Dr. Dobb's Journal

SOFTWARE TOOLS FOR ADVANCED PROGRAMMERS

#108 OCTOBER 1985 $2.95 ($3.95 CANADA)

The Future of Forth

Forth on a Chip

Forth on a Mac:
MacFORTH & Neon

nroff Hyphenation
on a Micro

MSDOS Quirks

Hobbles on Handles

10
0
38351 16562

H.E.L.P. Eliminates Every Bug known to Compilers . . . As well as a few other species

H.E.L.P. is a completely interactive C programming environment with three innovative full-sized features that will revolutionize the way you write code.

A Clean Compile — Guaranteed!

Say Good-bye to all compiler-type errors. **H.E.L.P.** 's built-in program checker (which would embarrass LINT) not only hunts down bugs . . . explains the proper syntax . . . gives examples of usage . . . but will even offer suggested corrections. If you want, **H.E.L.P.** will even make the corrections for you . . . at the touch of a key.

H.E.L.P. also finds semantic errors as well as poor style and inefficiencies. You can even check the portability of your code!

Multi-Window Editing

Open as many windows as you want . . . . there's no limitation . . . not even your own memory. Because **H.E.L.P.** uses a virtual memory system you can create programs larger than your machine capacity.

H.E.L.P.'s very powerful editor allows you the flexibility to work in several windows . . . with several files at the same time.

Save Keystrokes

Hundreds of commands are bound to the keyboard to give you fast execution.

Always be in Control

Not only can you develop code in many windows at the same time, but you can show (and refer to) important definitions in one window while creating in another. Or open a window and keep notes about your program . . . or type a memo . . . or a letter.

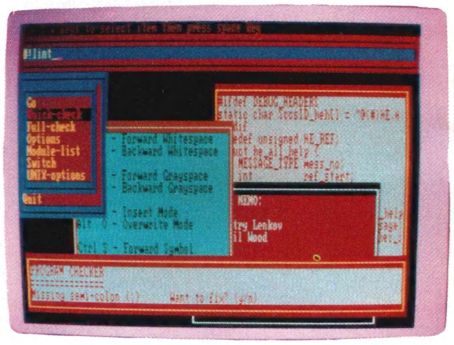
Increase your Productivity by 300% or More . . . . .

If you are a novice programmer, you'll begin writing code like an advanced programmer much faster with **H.E.L.P.**

Just imagine what **H.E.L.P.** will do for the ADVANCED PROGRAMMER.

You'll have more time to become creative with your algorithm (since **H.E.L.P.** will make sure your code compiles the "first time at bat").

H.E.L.P. tracks every step you make. If you are not sure about a command, just press a key, and you'll get the kind of help you need.



```
PROGRAM CHECKER
---------------
Missing semi-colon (;)        Want to fix? (y/n)
```

Check These Features

- *Multi-window environment*
- *Interactive program checking*
- *Check syntax, semantic, type usage, intermodule inconsistencies and portability*
- *Multi-file editing*
- *Intelligent help subsystem*
- *User-definable keyboard bindings*
- *Supports color and monochrome*
- ***H.E.L.P.** supports the full C Language*

NOW IN MS-DOS

EVEREST SOLUTIONS

Order now $395

Everest Solutions, Inc.
3350 Scott Boulevard
Building 58
Santa Clara, CA 95051
(408) 986-8977

Circle **no. 30** on reader service card.

New Version

Sizzling C.

The fastest C. The C that Microsoft developed to write its own software programs. Hot.

So hot that we can make this claim: Virtually every program runs faster with Microsoft® C Compiler than with any other MS-DOS® C compiler.

Efficient C.

We give you everything you need to write code so tight your computer will scream.

"Preliminary testing on the Microsoft C Compiler produced code that was significantly smaller than that produced by other C compilers."
Paul Springer, Ashton-Tate.

"We found the FAR pointer very helpful for situations where a mix of memory models offers the greatest efficiency."
Robert Frankston, Software Arts.

"The portability of the code between MS-DOS and XENIX® is great."
Jim Bean, Peachtree Software.

But it's really no surprise that our C stretches your micro to its limits. We wrote both the MS-DOS and the XENIX operating systems.

For the name of your nearest Microsoft dealer, or to upgrade from Microsoft C Compiler or Lattice C, call (800) 426-9400. In Washington State, Alaska, Hawaii and Canada, call (206) 828-8088.

MICROSOFT®
The High Performance Software™

And make your programs really cook.

Microsoft C Compiler Version 3.0

Microsoft C Compiler
- Produces compact code and fast executables.
- Implements register variables.
- Small, Medium and Large Memory model Libraries-Mix models with NEAR and FAR pointers.
- Transport source and object code between MS-DOS & XENIX operating systems.
- Library routines implement most of UNIX System V C library.
- Choose from three Math libraries and generate in-line 8087/287 instructions or floating point calls.
  - —Floating point emulator (utilizes the 8087/287 if installed).
  - —8087/287 coprocessor support.
  - —Alternate math package—extra speed on systems without an 8087/287.
- Link routines written in Microsoft FORTRAN (V 3.3 or higher), Microsoft Pascal (V 3.3 or higher) or Microsoft Macro Assembler.
- Supports MS-DOS pathnames and Input/Output redirection.
- File sharing and record and file locking is supported.
- Do source level debugging, with the Symbolic Debug Utility, available separately with the Microsoft Macro Assembler Package.*

Library Manager
Create, organize and maintain your object module libraries created with Microsoft languages.

Object Code Linker
- Simple overlay linker combines relocatable object modules created using Microsoft Languages into a single program.
- Link very large programs (over IMB, using overlays).

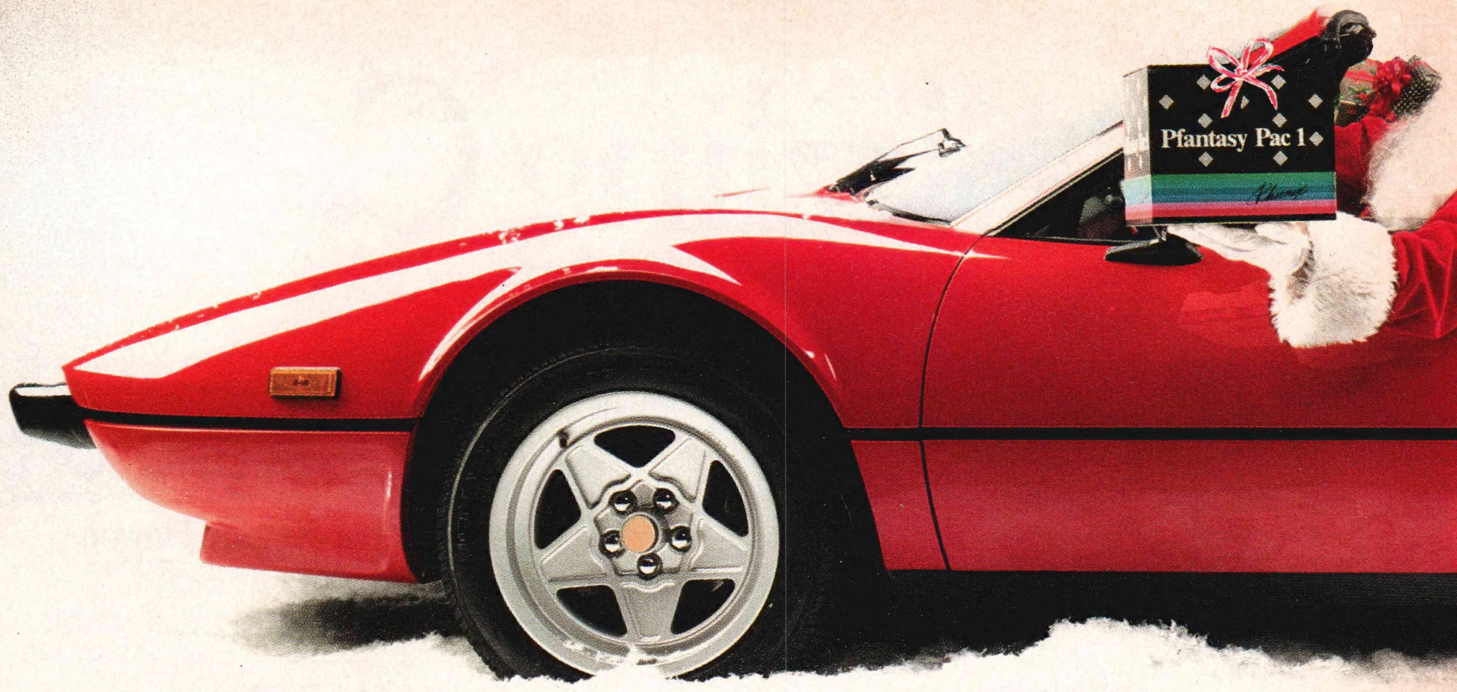
EXEPACK Utility
A new utility to compress sequences of identical characters from an executable file and to optimize the relocation table.

EXEMOD Utility
A new utility used to modify the fields in the header according to the instructions given by the user in the command line.

'C' Benchmarks—done on a Compaq Plus with 512k memory with no 8087. Program "SIEVE," with register variables,

	Exec Time	Code Size	EXE Size
Microsoft C	:9.39	141	5,914
Lattice C	:12.24	164	20,072

***Purchase both Microsoft C Compiler and Microsoft Macro Assembler and get a $25 rebate direct from Microsoft. See package for details.**



Buy Yourself A Pfantas

Just $~~129~~

Give your favorite programmer a holiday treat – Phoenix Pfantasy Pac1. Six of the best-engineered, highest performance programming packages on the market today. $1800 sold separately. $1295 as Pfantasy Pac1 with new software, and a unique on-line software update service. All at the one-time holiday price of $995. The perfect gift for the knowledgeable programmer. You'll get:

...Plink™86 Plus, our new linkage editor, adds a new dimension to modular programming...

Cache overlays to reduce response time. Link programs much larger than possible with standard linkers. Merge object modules so you can package them together. Reload overlays automatically, so it's easier to design program structures. Or, create up to 4,095 overlays, nested up to 32 levels, in one or many files; on one or many disks.

...Plus, Pmate,™ the text editor that works the way you do...

You can run Pmate in background while you perform another task, or call it up instantly while you're inside another program. With Pmate, you can reassign any command or key, so you can use keystrokes learned on other editors or make your keyboard Dvorak. And, you can assign complicated editing tasks to a keystroke, so you never have to re-key them.

...Pmaker,™ the perfect program development manager...

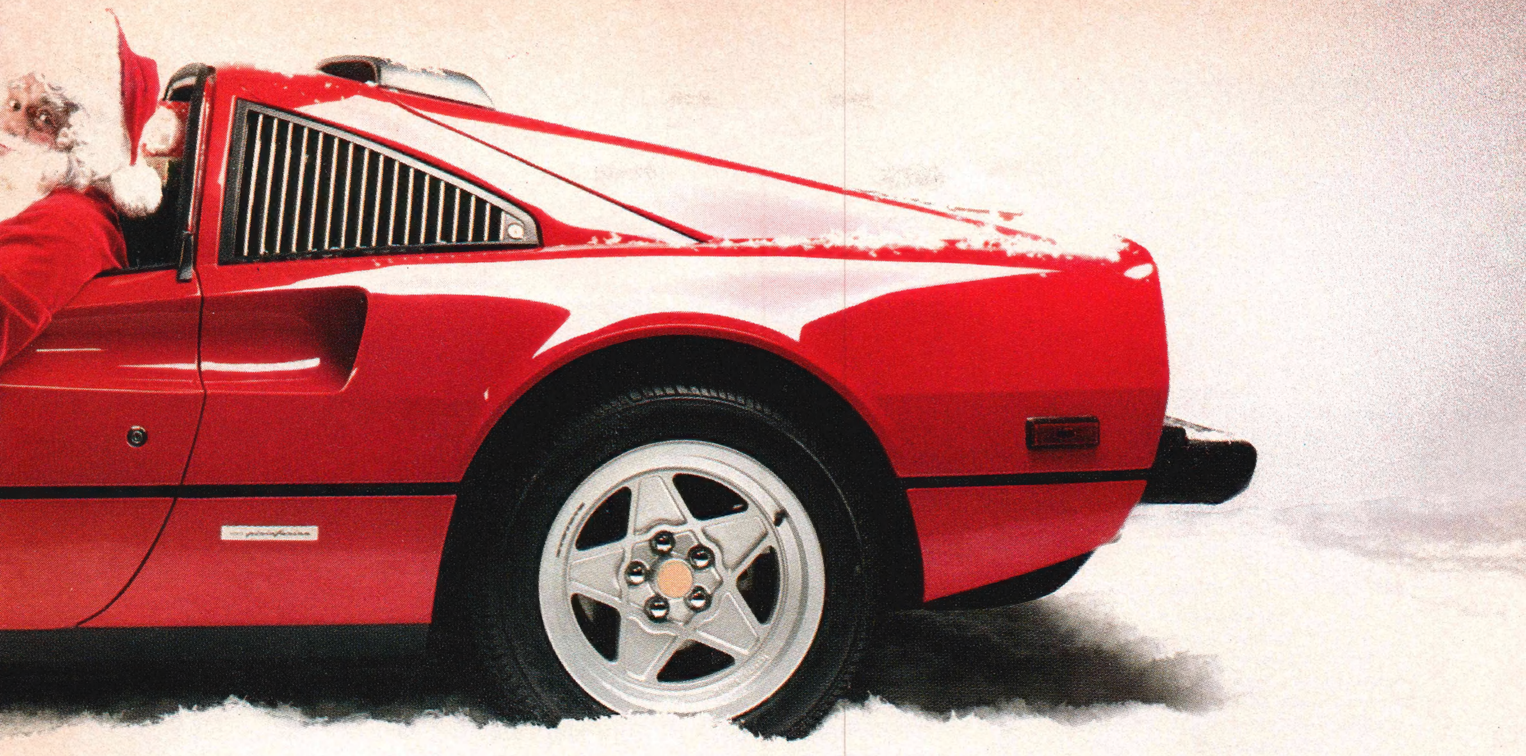
Like UNIX™ Make and MMF, Pmaker updates and tracks modifications in your programs. With Pmaker, generating a final copy of your program is never more complicated than typing PMAKER MYFILE, regardless of the number of modules that make up MYFILE, their dependencies on one another, the number of modules changed, or the variety of compile/assemble/ link steps required to generate the program.

...Pfix™86 Plus, the most advanced symbolic debugger on the market today...

With its adjustable window display of source file, disassembled object, data area, stack, breakpoint settings, CPU and coprocessor registers, Pfix takes the guesswork from debugging. You can debug without a listing, since you can see and enter symbolic names or absolute addresses in breakpoints, data displays, expressions, or with the in-line assembler. The entire 80xx and 80xxx series of

*Price effective through December 31, 1985.

Circle **no. 47** on reader service card.



y Pac™ For The Holidays.

~~5.~~ $995.*

16-bit processors is supported, along with the user-assignable string and numeric variables, up to 100-step trace-back, and debug log to disk or printer. Breakpoints can be set in the source file window, whose display can be synchronized with the disassembled object during tracing; symbols can be added incrementally during the debug session; and the disassembly written to disk.

...Pfinish,™ the program that maximizes your program's efficiency...

Pfinish helps you turn your beta-test product into a software work of art. It analyzes your program or the entire operating environment during execution, and produces reports which tell you how much time was spent in each routine or interrupt, who called it, how many times, and much more. Wasteful, inefficient, or non-optimal areas of code become immediately apparent, whether they're in your code, the compiler library, or in the operating system routines themselves.

...Ptel™ gets the lead out of binary file transfer...

You get error-free file transfer and access to mainframes, minis, and micros. ASCII. XModem. Modem7. Telink and Kermit. MS-DOS® 2.x or 3.x. With Ptel. You can transfer 8-bit binary files over a 7-bit data path with Kermit. Masked file-name transfers with XModem. Or, transfer whole subdirectories with a single command using Telink. Ptel keeps track of the original file size and creation date. Ptel also offers a script language, backward scrolling, and the ability to handle DOS commands from inside the program.

SEE US AT FALL COMDEX

...And, Phoenix's new on-line software update service. Free to all registered Pfantasy Pac1 owners.

Pfantasy Pac1 software works in MS-DOS/PC DOS environments, is completely language-independent, and is available for the IBM® PC, XT,™ AT,™ and compatibles, the Wang® PC, the TI Business-Pro,™ and the Tandy® 2000.

So, spoil yourself a little. Get Pfantasy Pac1, and make your favorite programmer's dream come true.

For more information contact: Phoenix Computer Products Corporation, 320 Norwood Park South, Norwood, MA 02062. Or call: (800) 344-7200. In Massachusetts (617) 762-5030.

Programmers' Pfantasies™ by

Phoenix

Programmers' Pfantasies, Pfantasy Pac, Pmaker, and Pfinish are trademarks of Phoenix Computer Products Corporation. Pmate, Pfix86 Plus, Plink86 Plus, and Ptel are trademarks of Phoenix Software Associates Ltd. MS-DOS is a registered trademark of Microsoft Corporation. Wang is a registered trademark of Wang Laboratories, Inc. Tandy is a registered trademark of Tandy Corporation. UNIX is a trademark of AT&T Bell Laboratories. IBM is a registered trademark, XT and AT are trademarks of International Business Machines Inc. TI Business-Pro is a trademark of Texas Instruments Incorporated.

Dr. Dobb's Journal

Editorial
Editor-in-Chief *Michael Swaine*
Managing Editor *Frank DeRose*
Technical Editor *Alex Ragen*
Editorial Assistant *Sara Noah*
Contributing Editors *Robert Blum, Dave Cortesi, Ray Duncan, Allen Holub*
Copy Editor *Vince Leone*
Typesetter *Jean Aring*
Production
Production Manager *Bob Wynne*
Art Director *Shelley Rae Doeden*
Production Assistant *Alida Hinton*
Cover Artist *Paul Ambrose*
Circulation
Sub. Fulfillment Mgr. *Stephanie Barber*
Subscription Mgr. *Maureen Kaminski*
Book Marketing Mgr. *Jane Sharninghouse*
Single Copy Sales Mgr. *Stephanie Barber*
Administration
Finance Manager *Sandra Dunie*
Business Manager *Betty Trickett*
Accounts Payable Supv. *Mayda Lopez-Quintana*
Accounts Payable Asst. *Denise Giannini*
Billing Coordinator *Laura Di Lazzaro*
Accountant *Marilyn Brown*
Adm. Coordinator *Kobi Morgan*
Advertising
Advertising Director
Shawn Horst *(415) 424-0600*
Systems Manager *Ron Copeland*
Administrative Manager *Anna Kittleson*
Advertising Sales
Walter Andrzejewski *(617) 567-8361*
Lisa Boudreau *(415) 424-0600*
Beth Dudas *(714) 643-9439*
Michele Beaty *(317) 875-0557*

M&T Publishing, Inc.
Chairman of the Board *Otmar Weber*
Director *C.F. von Quadt*
President *Laird Foshay*

Dr. Dobb's Journal (USPS 307690) is published monthly by M&T Publishing, Inc., 2464 Embarcadero Way, Palo Alto, CA 94303, (415) 424-0600. Second class postage paid at Palo Alto and at additional entry points.

Address correction requested. Postmaster: Send Form 3579 to *Dr. Dobb's Journal*, P.O. Box 27809, San Diego, CA 92128. **ISSN 0278-6508**

Subscription Rates: $25 per year within the United States, $46 for airmail to Canada, $62 for airmail to other countries. Foreign subscriptions must be pre-paid in U.S. Dollars, drawn on a U.S. Bank. For subscription problems, call: outside of CA 800-321-3333; within CA 619-485-9623 or 566-6947.

Foreign Distributor: Worldwide Media Service, Inc., 386 Park Ave. South, New York, NY 10016, (212) 686-1520 TELEX: 620430 (WUI)

People's Computer Company
Dr. Dobb's Journal is published by M&T Publishing, Inc. under license from People's Computer Company, 2682 Bishop Dr., Suite 107, San Ramon, CA 94583, a non-profit, educational corporation.

October 1985
Volume 10, Issue 10

CONTENTS

In This Issue

Once again we present our annual Forth issue. Perhaps the most significant development in Forth over the past year has been the direct implementation of Forth engines in silicon logic. One such silicon Forth engine is the Novix NC4000 chip. In our feature article Leo Brodie describes some of the revolutionary features of this chip and the benefits that it offers to programmers.

This month we also provide a solution to the problems of developing tight, rapidly executing code for PROM-based applications written in Forth. H. Robinson, P.D. Morse and S.A. Bowhill of the Central Illinois Chapter of the Forth Interest Group have designed a post-compiler that selects from compiled, threaded Forth code the segments necessary for the execution of the PROM-based application and redirects to the RAM of the target machine any code that must be resident there.

Our review section this month also focuses on Forth, in particular, on Forth and the Macintosh. First of all, Bruce Horn examines Neon, an object-oriented programming environment that is "a cross between Smalltalk and Forth." Then, Alan Clute takes a look at MacFORTH, a development package designed to ease the writing of programs in the Mac environment.

Our columnists have added their usual monthly goodies. Dave Cortesi serves up some further tidbits from his study of MSDOS, showing how to handle ambiguous pathnames with DOS functions 4Eh and 4Fh, how to define MSDOS diskette formats and how to control Control-Z.

Allen Holub had hoped to code a version of the original $\mathrm{T_EX}$ hyphenation algorithm in C for last month's issue, in which we reviewed two implementations of $\mathrm{T_EX}$ for the IBM PC. Limitations on his time prevented him from realizing his intentions before deadline, but he did manage to finish off the project for this month's C Chest. So, if you want to find out how to hyphenate supercalifragilisticexpialidocious . . .

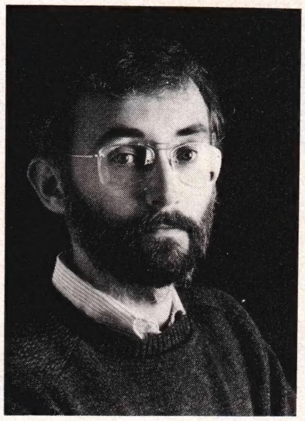
Bob Blum seems to be gleefully drowning in the deluge of new CP/M products that developers are sending to him. This month he finishes his discussion of the AMPRO Little Board, reviews SLRMAC, the 8080 assembler from SLR Systems, and offers some enticing glimpses of Date Stamper from Plu*Perfect Systems, DSD80, the debugger from Soft Advances and the latest single board computer from Micro Mint.

Finally, we continue this month in Mac Toolbox the program listings for the MacSCSI hard-disk interface presented by John Bass last month. Look for more on the MacSCSI next month.

Dr. Dobb's Journal

ARTICLES

EDITORIAL



Charles Moore first created Forth to control astronomical instruments; lately his attention has been focused on the design and implementation of a microprocessor that uses Forth as its machine language. The Novix chip represents only one recent effort to put Forth on silicon. Rockwell fit a good chunk of Forth into its R65F11 chip, along with the 6502 instruction set. The British Metaforth MF16LP single-board computer, which is implemented with custom bipolar circuits, uses Forth as its machine language. Bipolar technology is also incorporated in the H4TH/X Forth engines from Hartronix, which provide a real-time system with up to 4,000 primitives in firmware.

None of these products (with the exception, perhaps, of the H4TH/20) is of much practical significance to the Forth programmer developing commercial software on and for personal computers. They are more likely to interest programmers working in areas similar to those worked on by the first Forth programmer: device control, process control, robotics. Forth on a chip can only help speed developments in these fields.

There are grounds for wondering whether the programmer who uses Forth to develop personal computer applications is in a dwindling minority. The virtues of Forth in such applications are similar to those of C. Are Forth's additional virtues the kind that will secure it a permanent place in personal computer software development? Perhaps they are, but doubts in that area may have been one of the reasons why the language Neon was developed as an extension of Forth. Even so, extending Forth to the point where it becomes a different language is not a new idea.

Forth seems in excellent health, but hardware Forths and extended Forths lead one to wonder in just what area Forth will see its greatest use in the next year, and even whether the language will soon cease to be regarded as a high-level language. One wonders: Where is Forth bound?

Where *DDJ* is bound is, we hope, evident from its ten-year trajectory: on a path of providing more software tools for advanced programmers. We can be more specific. Here is our 1986 editorial calendar.

January: 68000 Programming. Article deadline: past.
February: Pascal, Modula-2, Ada. Article deadline: now.
March: The Future of Programming: algorithms for parallel processing. Article deadline: 12/1/85.
April: Artificial Intelligence. Article deadline: 1/1/86.
May: At the Human Interface: designing usable software. Article deadline: 2/1/86.
June: Communications. Article deadline: 3/1/86.
July: Forth. Article deadline: 4/1/86.
August: C. Article deadline: 5/1/86.
September: Algorithms. Article deadline: 6/1/86.
October: 80286/80386 Programming. Article deadline: 7/1/86.
November: Graphics. Article deadline: 8/1/86.
December: Operating Systems. Article deadline: 9/1/86.

Plus, of course, our regular columns on C, Unix, 16-bit software, CP/M and Macintosh software development tools.

Michael Swaine

Michael Swaine

Software Development Tools

C
Pascal
PLM
Assemblers
Simulators
VT-100 Emulator
CPM/80 Emulator

If:

Manufacturer-Compatible, Fully-Supported Cost-Effective, Cross and Native Development tools are important to you!

Then:

Let us tell you about MICROTEC RESEARCH's extensive line of field-proven software development tools for serious software developers.

Manufacturer Compatible

MICROTEC RESEARCH has been providing flexible and economical solutions for software developers since 1974. We've grown with the industry as our cross software development tools have transformed many large and small computers into powerful cross development systems for microprocessor applications. Our software tools are manufacturer compatible. Therefore, software made using manufacturers development systems may be more productively used in the MICROTEC RESEARCH cross-development environment.

Fully-Supported

MICROTEC RESEARCH's products are continually maintained and updated based upon the experience of thousands of installations worldwide. Revisions and modifications are distributed to licensees in warranty or covered by a service contract. Upgrades, which are major enhancements to a product's capabilities, are available at a significant discount. Our update/upgrade policy assures users they'll always be current with the fast moving world of microprocessor development.

Effective Differences

Beginning with product concept, through development, quality assurance, and post-sales support — **Quality, Compatibility,** and **Service** are the differences which set MICROTEC RESEARCH apart from the others.

Start Saving Time & Money

If you are a serious software developer, shopping for software development tools, call or write today for more information. **800-551-5554** In CA call **(408) 733-2919**

Target Microprocessors		Host Computers	Software Tools
8086/80186	68000/08/10	DEC VAX, PDP-11	C Compilers
8096	6809	DG MV-Series	Pascal Compilers
8080/8085	6800/01/02	DG Eclipse	PLM Compiler
8051	6805	Apollo	Assemblers
8048	6301	UNIX Systems	Linking Loaders
Z8002	6305	IBM PC	Librarians
Z80	6500	Data General/ONE	Download Communications
Z8	6500	HP 150	VT-100 Emulator
NSC 800	G65SCXX, G65SC1XX	DEC Rainbow	CPM/80 Emulator
9900...	R65C00, R6500/1...	others	
	others		

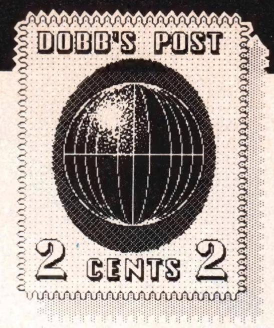
for Serious Software Developers

MICROTEC RESEARCH™

3930 Freedom Circle, Suite 101, Santa Clara, CA 95054
Mailing Address: P.O. Box 60337, Sunnyvale, CA 94088
(408) 733-2919 • Telex (ITT) 4990808

MICROTEC is a trademark of Microtec Research, Inc. Santa Clara, CA

Circle no. **50** on reader service card.



Hardware

Dear *DDJ*,

I vote for more and continuing hardware coverage, construction and otherwise. Every magazine except *BYTE* seems to be doing less and less with it. Yet, over time, Steve Ciarcia's hardware features are consistently among the most popular *BYTE* publishes.

The print spooler [in the July, 1985 issue of *DDJ*] can be simplified considerably if 6264 or similar 8K × 8 static RAM is used instead of 4164s. Seven 6264s would give 56K of RAM, leaving the top 8K of the 64K address space for ROM and the 6821 PIA. A 74LS138 3/8 decoder could break the address space into 8K chunks and supply the enable signals for individual devices. Additional decoding would be required only in the top 8K, containing EPROM and the PIA. That could be done with a single NOR gate. Two 74LS541 buffer packages could drive A0–A12, R/W, and the enable (phase 2) clock. This buffering is desirable due to the loading imposed by the additional 3-state outputs that use of 7 6264s would involve. Software could be much simpler because of elimination of the refresh routines. System clocks would also be simplified. Total cost of construction wouldn't be much more than the spooler presented.

The S-100 real-time clock article was interesting. However, there were a few errors and omissions that deserve comment.

The use of the MM58167A's RDY line to induce WAIT states depends on rather critical timing relationships among various events. These timing considerations can be greatly reduced if an interface adapter such as the 8255 is used to control the clock chip. The only critical timing relationships that exist then are those involving the 8255's interface with the bus. Since it is much faster than the 58167 and its timing requirements more closely matched to system timing, fewer problems occur.

The POWER DOWN input needs to go low 1 microsecond before the power off only if the standby interrupt is connected to something. If this particular feature of the MM58167A clock chip is not being used at any time, the POWER DOWN input can be tied directly to +5 volt power. The 2N2222 and 2N2907 transistors and associated components can then be eliminated. In many cases the power circuit for pin 24 of the clock chip need be no more complex than two diodes, a resistor, a capacitor and a battery; however, if the clock is on a board that tends to be noisy, a more complex zener diode and transistor switch can be used there. An example of this can be seen in Steve Ciarcia's May 1982 *BYTE* article.

While the fastest fixed-interval interrupt frequency specified for the 58167 by National Semiconductor is 10 Hz, a 100 Hz interrupt can be obtained by selecting the counter-latch match (sometime called "alarm clock") interrupt address, then writing "don't care" states into the seconds/tens-of-seconds latches. A good deal of control can be obtained by writing valid comparison values into some of the other latches, and "don't care" into others. A "4" in the day-of-the-week latch and "don't care" everywhere else would get you 100 Hz interrupts all day every Wednesday. A "12" in the hours/tens-of-hours latches and "don't care" elsewhere gets you 100 Hz from 12:00:00 through 12:59:59 every day. Add the "4" in day-of-the-week and you get the 12:00:00 through 12:59:59 100 Hz interrupts every Wednesday. Other combinations of valid and "don't care" states get other patterns. For most people's purposes, this capability is, at best, somewhat interesting, but otherwise useless. For automatic data logging, however, it could be quite useful.

The GO command increments the counter if the seconds count is greater than 39, not 40, as stated on page 64.

It is frequently easier and more foolproof to use the seconds/tens-of-seconds counter reset address to start the clock at a known time. I think maybe the GO command was either dreamed up because there was an address that wasn't otherwise used, or else it happened by accident. A goof in a mask, perhaps? A bug that became a feature?

On page 62 it is stated: "Each January . . . the clock IC does not automatically change to a new year." While true, it is also misleading: the 58167 doesn't have years/tens-of-years counters (or latches). Any changing of the year gets done somewhere other than the 58167 clock circuit.

Someone who wanted to make a clock board that does have years counters could use an OKI MSM5832, National Semiconductor MM58274, or Motorola MCM146818 or MCM146819 or any other clock chip that tracks years.

Sincerely,
Frank Kuechman
8113 NE 25th Avenue
Vancouver, WA 98665

DDJ

The Ultimate Programmer's Editor
WENDIN'S XTC™

VERSION 3.0 STILL ONLY $99⁰⁰

SUPER PROGRAMMERS edit in XTC to make software development a snap! Just look at these powerful features:

MULTITASKING

XTC's built-in multitasking lets you run your macros in the foreground or independently in the background while you continue editing. A background process has full access to editor resources, and can be used to translate code from one language to another in REAL TIME, print files in the background, or even scan syntax while you type in code. Best of all, you can use XTC to edit source and documentation in any programming language!

COMPILE IN WINDOWS

All DOS compilers and utilities can be executed from within XTC using a single keystroke. While it runs, XTC captures your compiler's output and redirects it into your text, so you can compare compiler messages with your source code ON THE SAME SCREEN. And using XTC's macro language, Turbo Pascal is literally only a keystroke away. You can use other compilers and utilities inside XTC too — like Lattice "C," Microsoft Pascal, and IBM's Basic, to name a few.

MACRO LANGUAGE

XTC has the most powerful macro language in the editing world. XTC's macros aren't just keystrokes assigned to keys; they're real programs that can be used to automatically edit source code and data files. Like any real programming language, XTC has control structures like IF THEN ELSE, WHILE DO, REPEAT UNTIL, FOR NEXT, DUPLICATE N TIMES, INDEFINITE LOOP, EXIT, and BREAK LOOP. XTC also has INTEGER, BOOLEAN, and STRING variables to hold numbers, conditions, and pieces of text.

WINDOWS & BUFFERS

With XTC you can display up to 8 different files or parts of the same file on the screen at once. XTC's windows are programmable and can even be linked together to share files. XTC also has 20 other buffers that you can use to hold files and blocks of text.

WORDSTAR COMPATIBILITY

If you already know Wordstar commands, then you don't need to learn a new set of commands. If you want to customize XTC, just write macros to emulate the key layout you're used to. XTC can also read Wordstar files, and can even strip off all of the non-standard high bits with a single command.

LARGE FILE EDITING

XTC lets you edit files entirely in memory (using all available memory), or paged to disk, for maximum flexibility. You can choose how XTC buffers text.

INCREDIBLE EXTRAS!

- UNDO N TIMES
- REMOVE WORDSTAR HIGH BITS
- EDIT GRAPHICS DISPLAYS
- AUTOINDENTING MODE
- TAB EXPANSION/ COMPRESSION MODE
- EXTRA LONG LINES
- MACRO COMPILER
- TELEPHONE SUPPORT
- 150 PAGES OF DOCUMENTATION
- RUNS ON IBM / PC, XT, AND / AT COMPUTERS (AND TRUE COMPATIBLES)

COMPLETE SOURCE CODE

XTC comes with 7,000 lines of source code jam-packed onto two DSDD disks. Includes 13 modules written in Pascal, and 2 assembly libraries you can use to access the PC's screen, intercept software interrupts (like INT 21H functions), allocate and deallocate memory, and load and execute programs. It's all included FREE for your recreation and enjoyment!

XTC outperforms any other programmable editor on all IBM/PC, /XT, and /AT computers (and true compatibles). If you want to feel the power of XTC before you buy it, just ask for our demo disk (only $10) and try it out. When you buy XTC, we'll knock 10 bucks off the price.

To get your copy of XTC now, order it over the phone — we can ship it the same day you call! Or, send in an order, just like this one:

XTC	**$99.00**
Macro Compiler	**FREE**
Shipping, Handling, Insurance	**3.50**
Want it COD? Add this	**1.90**

TOTAL IT UP, AND SEND IT QUICK!

INCLUDED FREE!
Macro Compiler with full source code in "C" · Translates Ada control statements into XTC macros. It even runs in XTC windows!

ORDER HOTLINE
509/235-8088
CREDIT CARDS WELCOME!

WENDIN T.M.

BOX 266
CHENEY, WA 99004

The people who make quality software tools affordable.

Ada® is a registered trademark of the U.S. Department of Defense. Turbo Pascal is a trademark of Borland, Inc. XTC is a trademark of Wendin, Inc.

Circle **no. 112** on reader service card.

DR. DOBB'S CLINIC

by D.E. Cortesi

Here's a batch of small items on MSDOS and PCDOS. The first is based on a note from Nick Hammond of Torrens, Australia. There, in a magazine called *Your Computer* and in a column not unlike this one, appeared the plaint of a puzzled user.. Sometimes, he said, his everyday programs would misbehave strangely. DEBUG would bomb or fail to write a good file; FORMAT would loop or report track zero bad when it wasn't; and so forth. The only common factors were that the failures occurred when there were resident programs in the system and that they could be fixed by booting without the resident programs. Sometimes they could be fixed merely by changing the order in which resident programs were loaded.

Dismal Memory Access

Hammond wrote to explain this puzzle. When IBM designed the PC, he tells us, they fitted a Direct Memory Access (DMA) chip from an 8-bit family. It can address only 64K. To accommodate the 1Mb address space of the 8088, they tacked on an external "page register" to establish which of 16, 64K "pages" the DMA chip would use. Because an 8088 "paragraph" is 16 bytes and a "page" is usually 256 bytes, we might do better to call these 64K DMA blocks, "chapters."

What we must never call them, Hammond says, is "segments." Unlike the 65K segments the 8088 can address, which may begin at any paragraph boundary, there are only 16 DMA chapters, and they fall exactly on the 64K boundaries of storage.

As a result, the DMA chip is physically incapable of transferring a block of data that crosses a chapter boundary in a single operation. "You have to transfer as far as the boundary, change the page register, then transfer the rest," says Hammond. "This would be fine if the firmware in the BIOS ROM handled the housekeeping, but it doesn't."

For example, if you use BIOS interrupt 13h to access a floppy disk and your data area crosses a chapter boundary—if any byte in your buffer except the first has an address that is divisible by 65,536—the BIOS will return an error. Programs that don't specifically check for this condition will fail intermittently, depending on their load address, and that depends on what resident programs were loaded ahead of time.

It's our impression that I/O done through MSDOS does not suffer this constraint—just one more reason to avoid ROM calls.

Control That Zee!

MSDOS function 40h writes up to CX bytes to the file named by the "handle" in BX and returns in AX the number of bytes written. Ordinarily, all the bytes will be written and AX will equal CX afterward.

Usually the reason DOS will report writing fewer than CX bytes is that the output disk is full and there is no place to put them. That's a disastrous error from which a program has few courses of recovery.

We recently discovered another cause—one that isn't documented so far as we are aware. If the output data include a logical end of file byte (Control-Z), and if BX names the standard output handle (0001), and if standard output has *not* been redirected to a file, DOS will stop writing at the Control-Z and return the count of bytes that preceded it.

This behavior makes sense in a way; it makes it easy to copy a disk file to the screen without displaying the superfluous garbage that may follow logical end of file. On that account, we'd expect DOS to behave the same whenever the target file was implemented by a character device driver, whether it was named by a standard handle number or not. We didn't test to see if it was that consistent, or whether it only applied the rule to the handles for the standard output, print, and auxiliary files, or whether it was just a glitch on handle 0001, or whether it varied with the device driver.

The more consistent DOS is, the worse the problems are. How is a program supposed to distinguish a full disk from an incidental Control-Z? The only way we can think of is to use the ioctl function to get the attributes of the file's device driver, and see if it's a block device or a character one. Wonderful.

One program that doesn't check is the MS Pascal 3.0 runtime library. If a Pascal program copies from standard input to standard output and if input is an edited keyboard line including Control-Z and output is the screen, it will abort with a disk-full error.

Medium Smart

Further gleanings from our study of MSDOS can be found in the Table (page 12), a compilation of the numbers that define MSDOS diskette formats. We pieced it together from IBM and Microsoft manuals.

The second line of the table is headed "format flag-byte." That's what we call the first byte of the FAT, the same byte that Microsoft and the illiterati of Boca Raton persist in calling the "media descriptor," thereby driving us straight up the wall. The word "media" is *plural*, dammit, and is in no way a synonym for "disk." The recording *medium* used with dis-

New lightning-fast ZBASIC zaps the competition.

It's hot. It's brand-new. And light years ahead of anything else.

It's ZBASIC. Written *for* programmers *by* programmers. (If you know BASIC—you know ZBASIC!) Now you can write a program exactly the same way on an Apple, an IBM, a Tandy, or any other major micro and port the source code. You only write the program *once* ...and it runs on all the major micros. The commands stay the same—regardless of the computer* (even graphic commands and disk I/0!).

ZBASIC. Starting now, it's the only language you'll ever have to know.

AVAILABLE NOW!

- Works the same on all micros
- Uses same commands—regardless of computer make
- Structured Basic—(spaghetti optional)
- Device-independent graphics (same graphic commands on all computers)
- 6–54 digits of precision (selectable by user)
- Built-in interactive editor and compiler—to compile and execute, just type "run."
- Choice of alphanumeric labels or line numbers

Not copy protected

One low price gives you everything—there are no hidden costs. Only $89.95 complete.

*subject to hardware limitations.

The finest implementation of the BASIC language for microcomputers!

•=YES N/A=Not available	ZBasic Interpiler 3.0	TURBO PASCAL 3.0	MBASIC compiler	True BASIC	BASICA Interpreter (IBM PC)
IBM and compatibles	•	•	•	•	•
Apple //e, //c (6502)	•	N/A	N/A	N/A	N/A
Macintosh	4th Qtr.	N/A	N/A	N/A	N/A
CP/M-80 2.2, 3.0	•	•	•	N/A	N/A
TRS-80 Mod I, III, 4, 4p	•	N/A	•	N/A	N/A
Direct commands	•	N/A	N/A	•	•
Maximum scientific digits of accuracy. (COS, SIN, ATN, LOG, EXP etc.)	6 to 54 selectable by the user	11 Binary BCD N/A	16	16	6
Device Indpdnt Graphics (same CMDS all graphic modes and computers)	•	N/A	N/A	N/A	N/A
SAME File commands all computers?	•	N/A	N/A	N/A	N/A
STRUCTURED: Labels, Functions, LONG IF etc.	•	•	N/A	•	N/A
Same editor commands all versions/computers	•	•	N/A	N/A	N/A
Sieve benchmark (Byte January 1983, 10 iter's)	13.7 sec.	14.1 sec.	14.9 sec.	261 sec.	2190 sec.
Shell-Metzner SORT (Sybex-BASIC for Scientist's and Eng. 2,000 5 char. strings)	19 sec.	28 sec.	71 sec.	194 sec.	2700 sec.
Executable Machine Lang. & approx. File size	12k	12k	32k	N/A	N/A
PRICE with BCD BCD=No rounding errors)	89.95	109.95	450.00	N/A	N/A
PRICE without BCD	89.95 BCD FREE	69.95	395.95	149.95	Comes with computer

ZBASIC™

The lightning-fast BASIC from Zedcor

To order use this coupon or call

1-800-453-4100

Utah Residents 1-800-662-8666; Alaska Residents 1-800-982-1500

Mail to: ZEDCOR
3438 N. Country Club Road / Tucson, AZ 85716

(Name) ____________________

(Address) ____________________ (Apt.)

(City) ____________________ (State) (Zip)

(Day Phone) ____________________

CREDIT CARD—MASTERCARD/VISA/AMEX C.O.D.

☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

EXPIRATION DATE: ____________________

☐	ZBASIC IBM PC/ Compatible (128K, MS DOS 2.0 or better recommended)	$89.95	$________
☐	ZBASIC Apple //e, //c (128K, Dos 3.3)	$89.95	$________
☐	ZBASIC CPM-80 (Z—80—2.2 or 3.0) (Provided on 5¼" KAYPRO II-SSDD Format)	$89.95	$________
☐	ZBASIC TRS-80 (48K) (Circle Model 1,3 or 4— 6.2)	$89.95	$________
☐	ZBASIC Macintosh (Delivery 4th quarter)	$89.95	$________
☐	DEVELOPERS— BUY ALL 5, SAVE $50. $449.75 VALUE	$399.75	$________
	Arizona Residents Add 5% Sales Tax		$________
	SHIPPING: U.S. add $5.00 per program		$________
	Foreign or C.O.D. add $10.00 per program (U.S. currency only)		$________
	TOTAL		$________

All benchmarks and accuracies apply to standard IBM PC with 8088. Other computers and/or CPU benchmark speeds, accuracies and object code file sizes will vary depending on the computer clock speed, operating systems and other factors beyond our control.

	5 1/4-inch					8-inch	
Nominal capacity	160K	180K	320K	360K	1.2M	250K	1.2M
Format flag-byte	FEh	FCh	FFh	FDh	F9h	FDh	FEh
Entries in root dir	64	64	112	112	224	68	192
Bytes per sector	512	512	512	512	512	128	1024
Sectors per cylinder	8	9	16	18	30	26	16
Cylinders per disk	40	40	40	40	80	77	77
Total sectors	320	360	640	720	2400	2002	1232
Reserved sectors							
for bootstrap	1	1	1	1	1	4	1
for FAT copies	2	4	2	4	14	12	4
for root dir	4	4	7	7	14	17	6
total reserved:	7	9	10	10	29	33	11
Available sectors	313	351	630	708	2371	1969	1221
Sectors per unit	1	1	2	2	1	4	1
Number of units	313	351	315	354	2371	492	1221
Bytes per unit	512	512	1024	1024	512	512	1024
Data capacity (Kb)	156.5	175.5	315.0	354.0	1185.5	246.5	1221.0

Table
Format data for a number of MSDOS diskette layouts. The "number of units," returned in DX by DOS function 1Ch, can be used to distinguish between layouts.

kettes is an iron or chromium oxide in a plastic binder, but the named byte doesn't describe it.

Whatever you call it, that byte is a poor guide to disk layout. The same flags can be found on 5- and 8-inch disks of very different formats, and the flag F9h appears on both 80-track diskettes and fixed disks. There is a better indicator: DOS functions 1Ch and 36h both return in register DX the total number of allocation units on a disk. Read across the third-from-last row of Table 1; you'll see that each format has a unique number of allocation units.

Hear No Evil . . .

. . . is the policy of Microsoft regarding MSDOS. We found a bug in DOS and wrote it up in these words: "DOS function 43h, AL=00 (return file attributes of path *DS:DX) returns impossible values and no error in the carry flag when given the path 'C:\.' Under DOS 3.0, despite absence of a carry error signal, function 59h returns partly-correct, partly-contradictory information, suggesting an error was recognized." We attached debug output to prove these points and mailed it all to Seattle.

It was a waste of time. Microsoft is not interested in trouble reports from civilians; they only talk to companies. Or so said Doug Boa of Microsoft in an earnest reply. "Microsoft does not support either of our operating systems (MSDOS and Xenix) directly," he wrote. "We do not sell them to the public, but only to the equipment manufacturers . . . [who] are responsible for all support functions." Therefore, Boa said, "even though you are reporting a problem with a function call general to all MSDOS implementations, you still need to direct the report to the equipment manufacturer."

We like to think we can read a tone of honest regret in those lines. If we wrote an OS, it would just *kill* us to have to turn back trouble reports because we'd know that, first, the people who report them are more likely to give up than to submit them a second time to their OEMs, and second, the OEM is all too likely to ignore the reports, lose them, reject them out of

Vitamin C

A Healthy Supplement For C Programmers

SCREEN I/O FEATURES INCLUDE:
- dBase-like atsay(), atget(), readgets()
- Input formatting via picture clause
- Unlimited input validation
- Individual field color/attribute control
- Field specific help messages
- Status line for added field specific prompts
- Right-to-left numeric input with floating dollar signs, asteriks, and commas
- Insert/delete, next/previous field, etc
- Current field highlighting
- Fast, easy, and bulletproof

PERFECT WINDOWS
- Automatic interface with all screen I/O functions
- Full overlay and restore
- Multiple virtual screens
- Full collision protection
- Automatically keeps I/O within window boundries
- Grow, shrink and move
- Word wrap & margins
- Print to or scroll any window without colision even if it is overlayed by another!
- Much, much more!

More than just another function library! A well planned, fully integrated programming system complete with pull down menus & user help system!

Vitamin C • $149.95
Includes more features than there is room to mention, all source code, manual, tutorial and sample programs. No royalties on your applications. For most MS-DOS C compilers! Soon for UNIX / XENIX ! Include $3 shipping, $6 for second day. Phone orders with MC/Visa welcome! Prices & features subject to change without notice. Call for other products.

CREATIVE PROGRAMMING
(214)243-6197
Box 112097
Carrollton, Texas 75011-2097

Circle **no. 32** on reader service card.

ignorance (or habit), or just sit on them for a few months before passing them on.

So let's give a moment's sympathetic thought to the Microsoft developers, sitting around wistfully hoping that somebody will write them a letter they'll be allowed to read. But if you find a bug in MSDOS or Xenix, save your time and postage. We might add that the sysops of the Microsoft forum on CompuServe will also reject any technical queries or trouble reports on MSDOS or Xenix.

Narrow Paths

MSDOS permits a program to control files entirely in terms of compact strings of characters called pathnames. Having to convert the name of a file from a compact, punctuated string like C:DROSS.DAT to the fixed fields of a file control block and back again was a major irritation of file work in CP/M and early MSDOS. With pathnames, programs talk to DOS in the same terms with which users talk to programs. That should make things easier.

And it does, so long as you work only with single files with explicit names. Get what should be a pathname from the user and terminate the input string with a null byte; point to the string and tell DOS, "Open it." DOS takes care of case conversion and of parsing the string into its many parts: drive letter, directory names, filename, extension. If the path describes anything that DOS can accept as a file, your program gets back a numeric "handle" and is off to the races. If it doesn't, DOS returns either error 2, "file not found," or error 3, "path not found." The first means that the disk letter (if any) and directory names (if any) were valid but the filename and extension couldn't be found in the (last-named or default) directory. Error 3 means that something went wrong before the filename and extension could be checked. We should be delighted that DOS can handle all this complexity for us, because it would be a real punctuation (a real #$?!%$*&!) to have to program it.

It's as easy to erase, create, or rename any single file from its pathname, or set its attributes or timestamp, as it is to open one.

Winding Paths

Complications arise when you want to handle ambiguous pathnames (ones in which wild cards are used to represent multiple files), especially if you want your program to be as flexible as DIR or COPY.

Let's create an example. Suppose that on drive B you have a diskette with a small hierarchy of files:

```
top.dat
mydir
   inner1.dat
   inner2.dat
```

And suppose that at some time you moved to drive B and made MYDIR the current directory:

```
A>b:
B>cd mydir
B>a:
```

That, by the way, is a point that's poorly explained by the DOS manuals we've seen. They don't make it clear that *each* drive has its own current directory. A reference to a file by drive, name and extension, as in the example

DISCOVER THE LANGUAGE OF ARTIFICIAL INTELLIGENCE

Not Copy Protected

PROLOG V

At last! A Prolog with enough muscle to handle real-world applications for UNDER $100! Discover why Japan has chosen Prolog as the vehicle for their "Fifth Generation Machine" project to design intelligent computers.

CHOOSE FROM TWO GREAT VERSIONS:

PROLOG V-Plus
$99^{95}

- ▫ More Than 100 Predefined Predicates
- ▫ Large Memory Model (to 640K)
- ▫ Floating Point Arithmetic
- ▫ 150-Page User's Manual and Tutorial plus Advanced Programming Documentation
- ▫ Co-Resident Program Editor
- ▫ Calls to Co-Resident Programs
- ▫ Text and Graphic Screen Manipulation

PROLOG V
$69^{95}

- ▫ 70 Predefined Predicates
- ▫ Small Memory Model
- ▫ Integer Arithmetic
- ▫ 122-Page User's Manual and Tutorial

UNBELIEVABLE UPGRADE POLICY
Tough decision? Buy PROLOG V *and upgrade to* PROLOG V-Plus *within 60 days for only the difference in price plus a handling charge.*

STANDARD FEATURES ON BOTH:

- ▫ Clocksin & Mellish-Standard Edinburgh Syntax.
- ▫ Extensive Interactive Debugging Facilities
- ▫ Dynamic Memory Management (*garbage collection*)
- ▫ Custom-Designed Binder and Slipcase

THE CHOICE OF UNIVERSITIES
Generous university site licenses and an excellent teaching tutorial and reference guide have made PROLOG V the choice of universities nationwide. Call for details.

Interpreter for MS-DOS/PC-DOS

PHONE ORDE
(619) 483-8513

NO RISK OFFER
Examine the documentation at our risk for 30 days. If not fully satisfied, return with disk still sealed for full refund.

☐ PAYMENT ENCLOSED $ ____________
CA residents add 6% sales tax
☐ CHARGE MY: ☐ MasterCard ☐ Visa
Card No. ____________ Exp. Date ________
Signature ____________
Mr./Mrs./Ms. ____________
(please print full name)
Address ____________
City/State/Zip ____________

PROLOG V-Plus	$99.95
PROLOG V	69.95
UPGRADE ONLY	40.00

Return factory diskette and $30 plus $10 Handling

SHIPPING:

$ 5.00	U.S.
7.50	Canada
10.00	Carribean, Hawaii Air
20.00	Overseas Air

COD Orders Not Accepted
15 day check clearance

CHALCEDONY SOFTWARE
5580 LA JOLLA BLV
SUITE 126 D
LA JOLLA, CA
92037

Circle no. 19 on reader service card.

```
A>type b:inner1.dat
```

means "look up the file named INNER1.DAT in the current directory of drive B." It doesn't mean the *root* directory of drive B; that would be

```
A>type b:\inner1.dat
```

Nor does it necessarily mean in the directory last named to CD; that might have been a new current directory for another drive.

Now consider all the ways that you can refer to the files on drive B using DIR. You can ask about all files by not supplying any operand,

```
A>b:
B>dir
```

which will list all files in the current directory. You can do the same, when another drive's the default, by naming the drive.

```
A>dir b:
```

You can ask for all the files in the *root* directory of the disk by specifying "root" with a backslash:

```
A>dir b:\
```

By convention of DIR and COPY such a reference to a directory is implicitly a reference to all the files in that directory. It saves your having to type *.*, as in

```
A>dir b:\*.*
```

or the equivalent double-dot back-reference,

```
A>dir b:..\*.*
```

You may refer to files in any directory by naming the directories along the path to the files,

```
A>dir b:..\mydir\..\top.dat
```

including redundant ones, as in that example.

Primrose Paths

There are a lot of useful programs with specifications that start off like this: "for every file that matches the first operand, the program will . . ." For consistency, such programs ought to interpret their command operands just as generally as DIR does. Anything less will just confuse the users: they'll come back complaining "DIR finds this file, why doesn't your program?"

The key to handling ambiguous pathnames lies in the DOS functions 4Eh and 4Fh, search for first and next match to path, respectively. Point to a pathname string that may or may not be ambiguous and call DOS; if there's (another) file that matches the pathname, DOS drops information about it into the current data transfer area. One returned item is the filename and extension, formatted as a string.

Unfortunately, functions 4E/4F aren't as general as DIR is. They do not assume *.* after a directory or drive letter, for instance.

Even when they are successful, they leave you with a riddle: how can you *open* the file you've discovered? You have the pathname you used for the search, a user-entered string like b:\mydir\*.dat, for example. You have the filename and extension returned by DOS, INNER2.DAT, for instance. Now you'd like to do something to the file you've found—open it or erase it or something—but you don't have enough information. You can't just point to the filename string and request an open. The file may reside in a disk or directory other than the current default, and DOS won't find it. The necessary qualifiers are in the search string, but you can't get at them without parsing the string. Parsing pathnames was the job that we so gladly turned over to DOS.

As you already suspect, we wouldn't be going on about this if we didn't have a solution. We found that we could use DOS function 4300h (return attributes of file using pathname) to tell the type of path a user has provided. With fair reliability, the user's path specification can be processed into two versions: a search path for use with functions 4E/4F, and a lead-in path to be prefixed to the filename string that those functions return.

The logic of it is explained at tedious length in the comments of Listing One (page 16), the source of a C function that does the job. This function, **fixpath()**, would be called once to initialize a path provided by the user. If it returns a zero, there are no files to be found down that path. Otherwise it has made two new paths, a search path and a prefix path. If the input path was, say,

```
b:..\
```

("all the files in the parent of the default directory of drive B"), **fixpath()** will return a search path of

```
b:..\*.*
```

and a prefix, or lead-in, path of

```
b:..\
```

However, if the input string was

```
b:\mydir
```

it would return a search path of

```
b:\mydir\*.*
```

and a lead-in path of

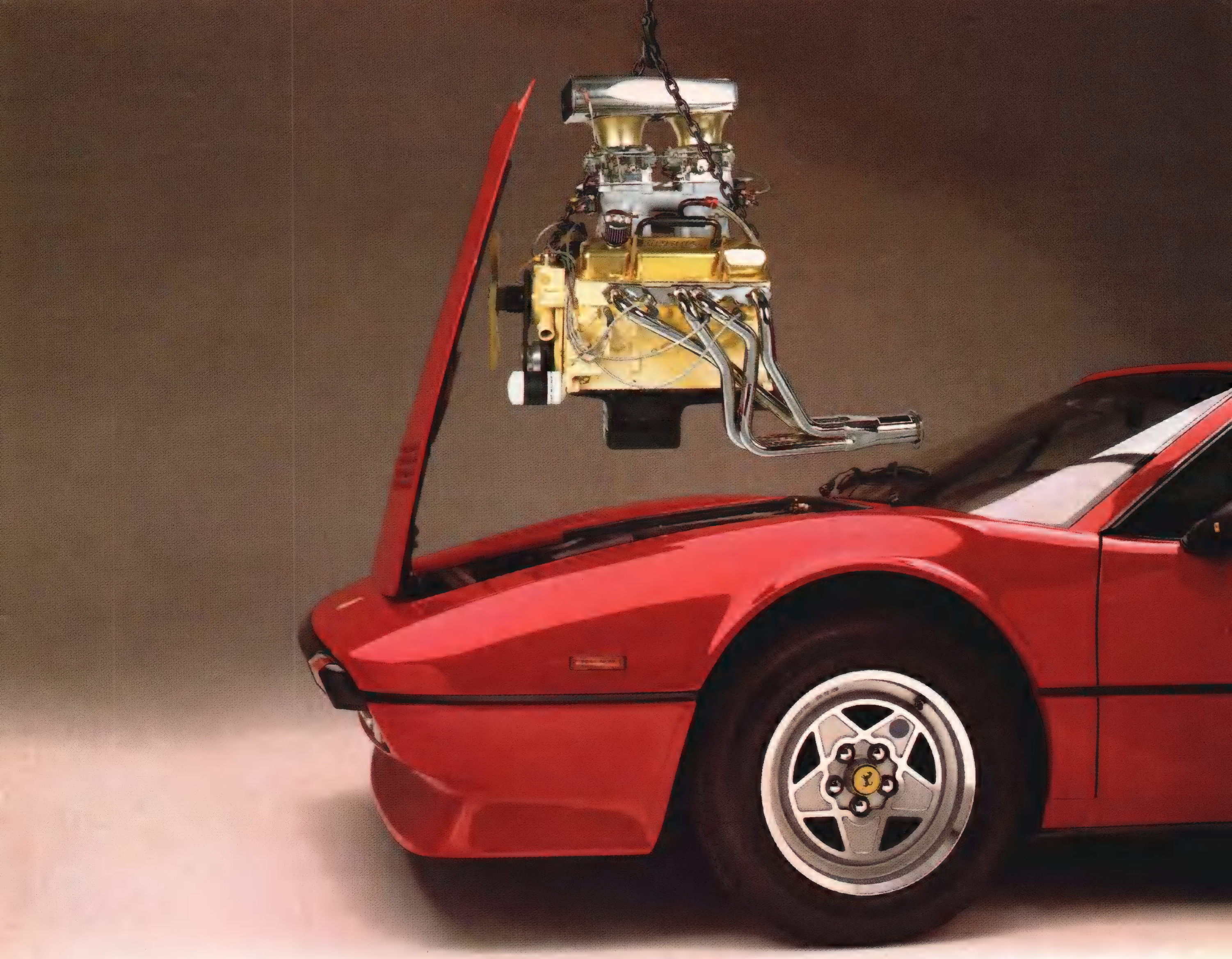
```
b:\mydir\
```

The search path is for input to functions 4E/4F. If function 4E returns an error, the original path was ungrammatical somehow. Otherwise, the search path will return the same filenames that DIR would print if it were given the same input path you gave to **fixpath()**. When you append one of these returned filename strings to the prefix path set up by **fixpath()**, you have a complete pathname to give to DOS functions 3Dh (open), 41h (erase), 43h (get/set attributes) or 56h (rename).

DDJ

(Listing begins on page 16)

Reader Ballot
Vote for your favorite feature/article.
Circle Reader Service **No. 190.**



AT™ Pfantasies for your PC or XT.™

Want better speed and memory on your PC or XT without buying an AT?

You've got it!

Phoenix's new Pfaster™286 co-processor board turns your PC or XT into a high-speed engine 60 percent faster than an AT. Three times faster than an XT. It even supports PCs with third-party hard disks. But that's only the beginning.

You can handle spreadsheets and programs you never thought possible. Set up RAM disks in both 8088 and 80286 memory for linkage editor overlays or super-high-speed disk caching. All with Pfaster286's 1mb of standard RAM, expandable to 2mb, and dual-mode design.

You can develop 8086/186/286 software on your XT faster. Execute 95 percent of the application packages that run on the AT, excluding those that require fancy I/O capabilities your PC or XT hardware just isn't designed to handle. Queue multi-copy, multi-format print jobs for spooling. Or, switch to native 8088 mode to handle hardware-dependent programs and back again without rebooting. All with Pfaster286's compatible ROM software. And, Pfaster286 does the job unintrusively! No motherboard to exchange. No wires to solder. No chips to pull. Just plug it into a standard card slot, and type the magic word, "PFAST."

If you really didn't want an AT in the first place, just what it could do for you, call or write: Phoenix Computer Products Corp., 320 Norwood Park South, Norwood, MA 02062; (800) 344-7200. In Massachusetts, 617-762-5030.

Programmers' Pfantasies™
by
Phoenix

XT and AT are trademarks of International Business Machines Corporation. Pfaster286 and Programmers' Pfantasies are trademarks of Phoenix Computer Products Corporation.
For the Ferrari aficionado: yes, we know this is a rear engine car. We are showing the addition of a second engine to symbolize how Pfaster can be added to your PC or XT to increase performance.

Circle **no. 91** on reader service card.

SEE US AT FALL COMDEX

FOR TRS-80 MODELS 1, 3, 4, 4P
IBM PC/XT, AT&T 6300, ETC.

THE COMPLETE FORTH GETS A MAJOR UPDATE: MMSFORTH V2.4

- A total software environment: custom drivers for printer, video and keyboard improve speed and flexibility. (New TRS-80 M.4 version, too!)
- Common SYS format gives you a big 395K (195K single-sided) per disk, plus a boot track!
- Common wordset (79-Standard plus MMSFORTH extensions) on all supported computers.
- Common and powerful applications programs available (most with MMSFORTH source code) so you can use them compatibly (with the same data disks) across all supported computers.
- Very fast compile speeds and advanced program development environment.
- A fantastic full-screen Forth Editor: Auto-Find (or -Replace) any word (forward or back), compare or Pairs-Edit any two ranges of blocks, much more.
- Temporary dictionary areas.
- QUANs, VECTs, vectored I/O, and many more of the latest high-performance Forth constructs.
- Manual and demo programs are bigger and better than ever!
- Same thorough support: Users Newsletter, User Groups worldwide, telephone tips. Full consulting services.
- Personal Licensing (one person on one computer) is standard. Corporate Site Licensing and Bulk Distribution Licensing available to professional users.

mmsFORTH

IT'S BETTER THAN EVER.

The total software environment for IBM PC/XT, TRS-80 Model 1, 3, 4 and close friends.

- Personal License (required):

MMSFORTH V2.4 System Disk	**$179.95**
(TRS-80 Model 1 requires lowercase, DDEN, 1 40-track drive.)	

- Personal License (additional modules):

FORTHCOM communications module	**$ 49.95**
UTILITIES	**49.95**
GAMES	**39.95**
EXPERT-2 expert system	**69.95**
DATAHANDLER	**59.95**
DATAHANDLER-PLUS (PC only, 128K req.)	**99.95**
FORTHWRITE word processor	**99.95**

- Corporate Site License Extensions . . . . **from $1,000**
- Bulk Distribution . . . **from $500/50 units.**
- Some recommended Forth books:

STARTING FORTH (programming)	**19.95**
THINKING FORTH (technique)	**15.95**
BEGINNING FORTH (re MMSFORTH)	**16.95**

Shipping/handling & tax extra. No returns on software.
Ask your dealer to show you the world of MMSFORTH, or request our free brochure.

MILLER MICROCOMPUTER SERVICES
61 Lake Shore Road, Natick, MA 01760
(617) 653-6136

Circle **no. 65** on reader service card.

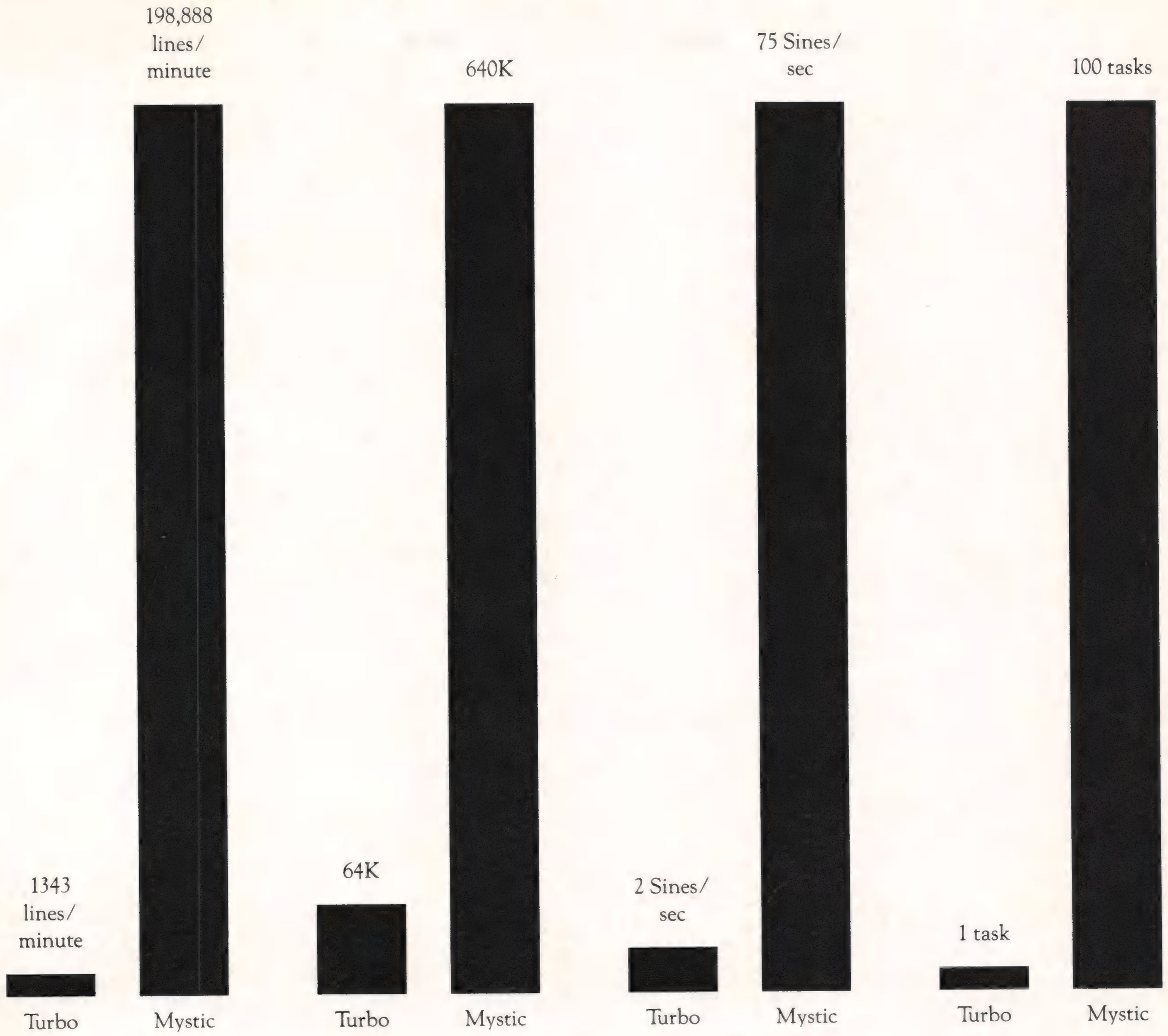
Dr. Dobb's Clinic Listing

(Text begins on page 10)

```
/* ===============================================================
   fixpath() processes a DOS pathname for two different uses.
The input path, *ip, is presumably a command operand like the
first operand of DIR.  One output, the search path *sp, is
the input possibly augmented with "*.*" or "\*.*" so that it
will work reliably with DOS functions 4E/4F.  The other output
is a lead-in path, *lip, which can be prefixed to the simple
filename.ext returned by functions 4E/4F to make a complete
path for opening or erasing a file.
   The function returns 1 if it is successful, but 0 if the
input path can't be processed and should not be used.
   Some input paths can be processed here yet be invalid or
useless.  The search path produced from such an input will
cause an error return from function 4E (search first match).
=============================================================== */
int fixpath(ip,sp,lip)
register char
        *ip, /* the input path */
        *sp, /* the search path */
        *lip; /* the lead-in or prefix path */
{
char *cp; /* work pointer for scanning paths */
/* ===============================================================
   Pick off 4 special cases:
        (1) the null string, which we take to mean "*.*"
        (2) the simple "d:" reference to a drive, which we
            also take to mean "d:*.*"
        (3) the root-dir reference "d:\" which is mishandled
            by function 4300 of both DOS 2.1 and 3.0.
        (4) any reference that ends in "\"
In all cases, the input path is ok as a lead-in, but it needs
the global qualifier *.* added for searching.
=============================================================== */
        if ( (strlen(ip)==0)                /* null string */
            ||(strcmp(ip+1,":")==0)         /* d: only */
            ||(ip[strlen(ip)-1]=='\\') )    /* ends in bkslsh */
        {
                strcpy(lip,ip);             /* input is ok prefix */
                strcpy(sp,ip);
                strcat(sp,"*.*");           /* add *.* for search */
                return(1);
        }
/* ===============================================================
   Ok, we have a non-null string not ending in \ and not a lone
drive-letter.  Four possibilities remain:
        (1) an explicit file reference,  b:\mydir\mydat
        (2) an explicit directory reference, \mydir
        (3) an ambiguous file reference, *.* or b:\mydir\*.dat
        (4) an unknown name, a:noway or b:\mydir\nonesuch.dat
We can separate this with fair reliability using DOS function
4300h, get attributes from path.
   The DOS operations used here are based on the MS-C 3.0
library functions, but all C compilers have similar library
routines and header files.
=============================================================== */
        regs.x.ax = 0x4300;      /* AH=43h, AL=00h */
        (char *)regs.x.dx = ip; /* DS:DX -> input path */
        regs.x.cx = 0;          /* clear CX */
        intdos(&regs,&regs);    /* call DOS */

        if ( (regs.x.cflag) && (regs.x.ax==0x0002) )
        {
        /* =======================================================
           The path is valid, in that all directories and drives
        named in it are valid, but the final name is unknown. No
        files will be found in a search, so quit now.
        ======================================================= */
                return(0);
        }

        if ( ((0!=regs.x.cflag)&&(regs.x.ax==0x0003))
            ||((0==regs.x.cflag)&&(0==(regs.x.cx & 0x0010))) )
        {
```

(Continued on page 18)

MYSTIC PASCAL vs. TURBO PASCAL



COMPILER SPEED

Mystic Pascal uses a completely new compiler design to give extremely fast compilation of ISO Standard Pascal programs. It compiles each line of code the instant you key it into the Full Screen Editor. (Bar Graph is based on compiling a 179 line program with Turbo 3.0 on IBM PC.)

MAXIMUM CODE SIZE

Mystic Pascal lets you use up to 640K for your program's code and data. Your Personal Computer cost thousands of dollars, why settle for a programming language that only lets you use one tenth of its power?

ARITHMETIC SPEED

Mystic Pascal's floating point arithmetic does not use an 8087 but runs almost like it did! We use a special way of storing numbers in memory to achieve single precision arithmetic that is 5 to 50 times faster than other compilers. If you still need 8087 support, we'll be offering that soon.

MULTI-TASKING

The Mystic Pascal system is based on multi-tasking. The Editor, Compiler and other components operate concurrently. Up to 100 of your Pascal procedures may also execute concurrently. And they may communicate by passing messages through queues.

MYSTIC PASCAL FEATURES

- Interactive Pascal
- EXE file output
- 8086 optimized code
- Full Screen Editor
- ISO Standard Pascal
- Help Windows for Pascal Language
- DOS INTeRrupt calls

Mystic Pascal requires an IBM PC or compatible with 256K. Turbo Pascal is a registered trademark of Borland International, Inc.

QUANTITY DISCOUNTS

The price below includes the disk, 75 page manual and shipping within the US and Canada. We can optionally ship by UPS Blue Label (2nd Day Air) within the U.S. Foreign orders add 10% shipping, minimum $15.

Quantity	Price Each	2nd Day Air Price Each
1	$64	$4.00
2- 4	56	3.00
5-10	48	2.50
11-20	42	2.00
21-50	38	1.50

USE YOUR CREDIT CARD TO ORDER NOW BY PHONE (505) 757-6344.

Mystic Canyon Software
P.O. Box 1010
Pecos, New Mexico 87552

Quantity ____________
Price ____________
Blue Label ____________
Total ____________

Name ______________________________
Address ______________________________
City ____________ State ______ Zip ____________

Sorry, CODs and Purchase Orders are NOT accepted. Payment must be in US funds on a US bank.

☐ Check/Money Order ☐ Visa ☐ Mastercard

Card ______________________ Exp. ____________
Signature ______________________________

Circle no. 79 on reader service card.

(Listing continued, text begins on page 10)

```
        /* ======================================================
           Error 3, path not found, could mean total junk or a
        misspelt directory name, but most likely it just means
        the path ends in an ambiguous name.  If there's an error
        the initial search (function 4Eh) will fail.
           With no error and reg cx NOT showing the directory
        attribute flags 0010, we have a normal, unambiguous file
        pathname like "b:\mydir\mydat" or just "myprog.com".
           In either case the search path is the whole input
        path.  The lead-in is that shorn of whatever follows the
        rightmost \ or :, whichever comes last -- or just a null
        string if there is no \ or :.
        ==================================================== */
                strcpy(sp,ip); /* working copy of ip in sp */
                cp = sp+strlen(sp)-1;
                for(; cp > sp; --cp)
                        if (('\\'==*cp)||(':'==*cp)) break;
                if (cp>sp) ++cp; /* retain colon/slash */
                *cp='\0'; /* may make a null string */
                strcpy(lip,sp);
                strcpy(sp,ip); /* whole input for search */
                return(1);
        }

        if ((0==regs.x.cflag)&&(regs.x.cx & 0x0010))
        {
        /* ======================================================
           No error and the directory attribute returned in cx
        shows an unambiguous path that happens to end in the
        name of a directory, for instance "..\..\bin" or
        "b:\mydir"  Applying the rules of DIR or COPY, we
        have to treat these as global refs to all files named
        in the directory.  The search path is the input with
        "\*.*" tacked onto it.  The lead-in path is just the
        input plus a backslash.
        ==================================================== */
                strcpy(sp,ip);
                strcpy(lip,ip);
                strcat(sp,"\\*.*");
                strcat(lip,"\\");
                return(1);
        }
        else
        { /* unexpected events make me nervous, give up */
                return(0);
        }
}
```

End Listing

Sequitur™

- **Relational Database with Word Processor**
- **Visual Programming Syntax-Menu and Example**
- **Runs on All MSDOS 2.X and 3.X Computers**

Golemics, Inc.
2600 10th St., Berkeley, CA 94710
(415) 486-8347

Circle **no. 45** on reader service card.

Get the **ProDOS**™ advantage for
+
all your **Aztec C65**™ programs

VIX is a UNIX like operating system designed to run Manx's Aztec C65 software under ProDOS. With VIX, programs running under the SHELL will run under ProDOS including c65, cci, as65, asi, ln, mklib and others. System includes:

- Standard utilities: cat, cp, date, l, mkdir, ren, rm and stty.
- An improved library written in 6502 assembly.
- A fast screen editor with undelete, auto-indent, work wrap and more.
- Source code to entire system except editor.

VIX - $49.95 + $3.50 shipping
Balanced binary tree data base library- $75
b-tree with source-$350

Eclipse ● Systems
223 Matthew Road
Merion Station, Pa. 19066
(215) 664-2419

Circle **no. 168** on reader service card.

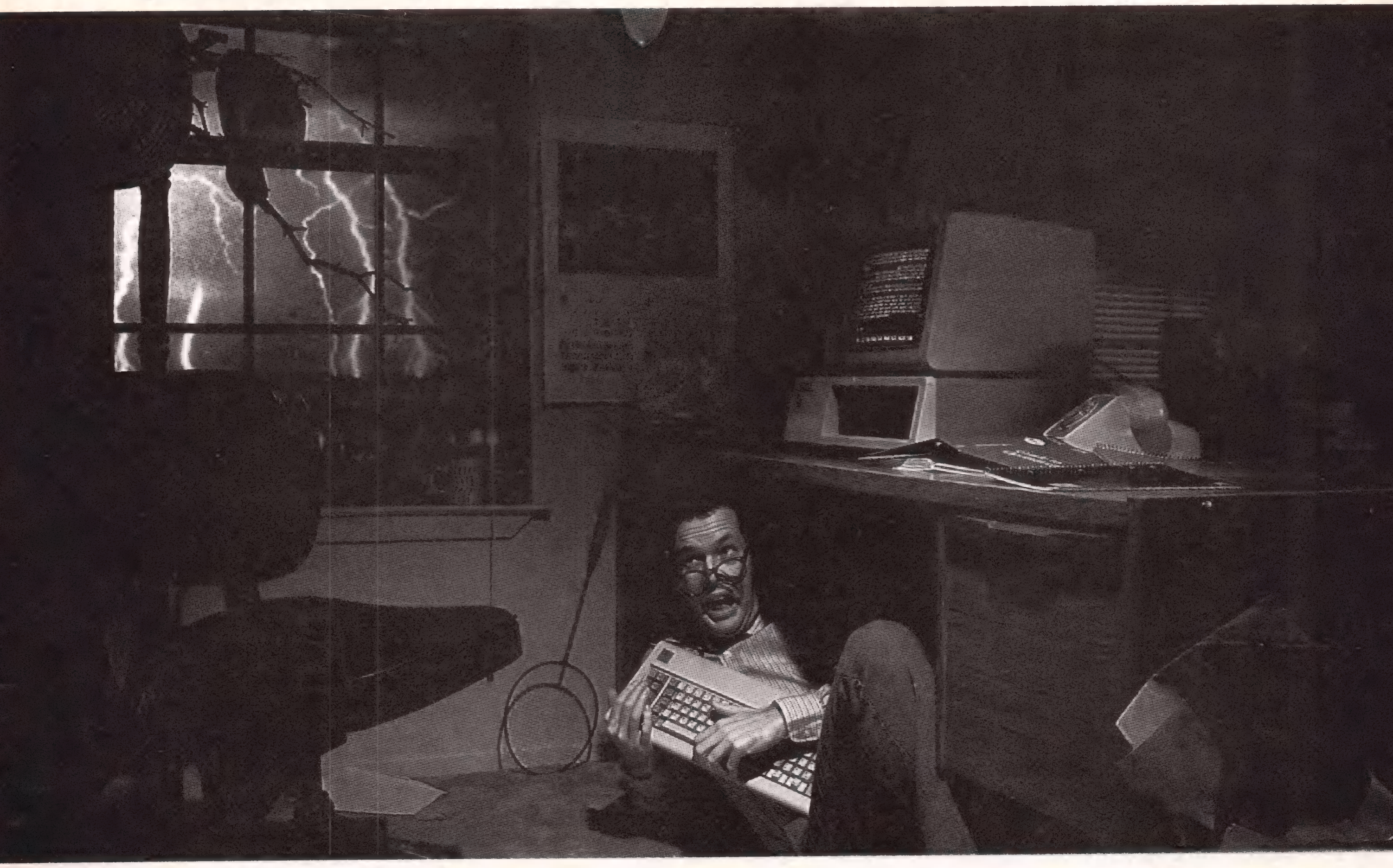
FORTRAN PROGRAMMERS

Discover why
you should be using
F77L
the complete implementation
of the ANSI FORTRAN 77
Standard for the IBM PC and
compatibles.
If you are serious about your
FORTRAN programming, you
should be using F77L.
$477

Lahey Computer Systems, Inc.
31244 Palos Verdes Drive West, Suite 243
Rancho Palos Verdes, California 90274
(213) 541-1200
Serving the FORTRAN community since 1969

Circle **no. 166** on reader service card.

If lightning still scares you, you're using the wrong file manager.



Be sure. Btrieve™.

Lightning may strike. But it doesn't have to destroy your database.

Btrieve™ file management offers automatic file recovery after a system crash. So accidents and power failures don't turn into database disasters. Your Btrieve-based applications will come up when the lights come back on.

Fast. Btrieve is lightning fast, too. Its b-tree file structure automatically balances—you never waste time reorganizing the index. And Btrieve is written in Assembly language especially for the IBM PC™. The result: electrifying speed on file maintenance routines. Applications that run fast. Users who don't waste time waiting.

The standard for networking. When your application requires multi-user file sharing, Btrieve/N (network version) sets the standard for the industry's most popular LANs: IBM's PC Network, Netware™, Davong MultiLink™, Omninet™, PC Net™, EtherSeries™, Nestar™, and NetOne™. Btrieve/N offers **safe** network file management that coordinates simultaneous updates and protects against lost data.

Fully-relational data management. Using SoftCraft's entire family of products gives you a complete, fully-relational database management system. Rtrieve™ adds report writing capabilities for generating the reports you need. Xtrieve™ speeds users through database queries with interactive, on-screen menus—no command language or special syntax.

For professional programmers. Btrieve is the fast, reliable answer for all your application development. In any development language—BASIC, Pascal, Cobol, C, Fortran, and APL. With multikey access to records. Unlimited records per file. Duplicate, modifiable, and segmented keys. Variable cache buffer.

With Btrieve, you can develop better applications faster. And know they'll be safe if lightning strikes.

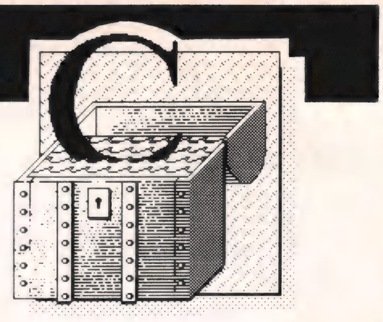
Suggested retail prices: Btrieve, $245; Btrieve/N, $595; Xtrieve, $195; Xtrieve/N, $395; Rtrieve, $85; Rtrieve/N, $175. Requires PC-DOS or MS™-DOS 1.X, 2.X, or 3.X.

Btrieve, Xtrieve, and Rtrieve; IBM; Netware; Davong MultiLink; Omninet; PC Net; EtherSeries; Nestar; NetOne; and MS are trademarks of SoftCraft Inc.; International Business Machines; Novell Data Systems; Davong Systems Inc.; Corvus Systems; Orchid Technology; 3Com Corp.; Nestar Systems Inc.; Ungermann-Bass; and Microsoft Inc.

SoftCraft Inc.
P.O. Box 9802 #917
Austin, Texas 78766
(512) 346-8380 Telex 358 200

Circle **no. 118** on reader service card.

C CHEST



by Allen Holub

This month we're going to look at an implementation of Knuth's hyphenation algorithm. This code is yet another part of my version of the Unix text formatter, nroff.

Bugs and Fixes

Before getting started, here's a little miscellany. First, while bringing up a version of **printf( )** the other day, I found a bug in Version 2.15 of Lattice C (though the developer would probably argue that it's a feature).

When writing a subroutine with a varying number of arguments of varying types (like **printf**), arguments are fetched from the stack frame by casting a "generic" pointer to point at an individual object on the stack. After fetching the argument, the pointer is advanced past it. In other words, we want to cast a character pointer into a long pointer, fetch a long object off the stack, and then advance the character pointer by the size of a long. The first thing I tried was:

```
char *p ;
long x ;

x = *(long *)p++ ;
```

but this won't work correctly. Here, the problem is order of precedence. *****, **(long *)**, and **++** are all at the same precedence level, but they associate right to left. When fully parenthesized, the associativity gives us:

```
x = *((long *)(p++))
```

Because the **++** binds more tightly to **p** than the cast, the compiler will assume that **p** is a character pointer (rather than a long pointer) for the sake of the increment, though it will fetch the long object correctly. In other words, the expression evaluates to a long, but 1 (rather than 4) is added to the pointer. I tried to force the precedence to behave by doing:

```
x = *( (long *)p )++ ;
```

but now Lattice gives me an error 25 (lvalue required) and won't compile it. By itself, **((long *)p)++** doesn't compile either. Because this statement indeed has an lvalue (the **x**), I was rather nonplused. I think that the compiler was confused by the cast, thinking that I was trying to do something such as **x++ = y** or:

```
int foo;
(long)foo = 5L;
```

which won't work because this would force **foo** to overflow (only an integer's worth of space is reserved for **foo**, and we're trying to squeeze a long object into it). However, **((long *)p)++** should work because we're just modifying the way that the pointer arithmetic works, not overflowing the target. A more clever error detection algorithm would take care of this exception. The following, though inelegant, does work:

```
char *p;

x = *(long *)p ;
p += sizeof( long );
```

While we're on the subject of bugs, several people have written about what we thought was a bug in **ls**, the directory utility printed a couple months back. The problem here is underlining. On a Compaq, directory names comes out at half intensity instead of underlined. The problem here is the hardware (and I contend that this is a bug, not a feature). The Compaq emulates an IBM color card running a monochrome monitor. The color card doesn't support underlining (I think that it should) but prints text in a different color rather than underlining it. On a monochrome display, this second color comes out as reduced intensity. All the IBM cards, color or otherwise, should support the attributes specified in the ANSI.SYS documentation. Anyway, the Paradise Modular Graphics card has the same problem as the Compaq. On the other hand, the Hercules Graphics card does underlining correctly.

There's a second bug in **ls** that's a little more serious. **Ls**, as it stands, runs fine under DOS Version 3.0. However, when running under Version 2.x, it couldn't find the root directory (\) when it was explicitly requested by an **ls ** command. A version-independent fix for the problem is given in Listing One (page 26).

Hyphenation

Moving on, the capability to hyphenate words is desirable in any text formatter or word processor. A lot of needless white space between words can be eliminated by hyphenation. Narrow columns look better, and we can get more words per page. Looking at the output of several commercially available word processors, I suspect that most of them derive their algorithms from the one given by Knuth in *T_EX and Metafont*, and I've used his algorithm here.[1]

Hyphenation is a harder problem than you would think at first. To use Knuth's examples, why is *record* hyphenated *rec-ord* when used as a noun and *re-cord* when as a verb? Consequently, it's impractical to attempt a truly universal algorithm—there are just too many exceptions. Knuth's algorithm is intentionally conservative—if it isn't absolutely sure how to split two syllables, it doesn't split them.

Algorithm Description

Knuth attacks the problem by breaking it into several parts, the first of which is exception removal. He keeps a dictionary of words that are known exceptions to his other rules and then sees if a word is an exception before processing it further. Strictly speaking, we could use a huge dictionary and dispense with the rest of the algorithm, but this would be too cumbersome to be practical. Knuth includes a dictionary of about 300 words in his book, some of which are rather useless. (How many times do you use "coneflower" in an average document?) There just wasn't room to print the dictionary look-up part of the hyphenator this month. It's a straightforward program—it reads a sorted list of exceptions and then does a simple binary search in a table.

The second part of the algorithm does suffix removal. The following suffixes are recognized:

> *-able, -ary, -cal, -cate, -cial, -cious, -cient, -dent, -ful, -ize, -late, -less, -ly, -ment, -ness, -nary, -ogy, -rapher, -raphy, -scious, -scope, -scopic, -sion, -sphere, -tal, -tial, -tion, -tional, -tive,* and *-ture.*

Some of these are qualified

> *-able* must be preceded by *e, h, i, k, l, o, u, v, w, x, y, nt,* or *rt*; *-ary* must be preceded by *ion* or *en*; *-cate* and *-late* must be preceded by a vowel (*a, e, i, o, u,* or *y*); *-nary* may not be preceded by *e* or *io*; *-cious* may not be preceded by *s*; and *-ize* must be preceded by *l.*

Another complication is *-ing.* Knuth says if *-ing* is preceded by fewer than four letters, don't insert a hyphen. If *-ing* is preceded by two identical consonants (other than *f, l, s,* or *z*) put the hyphen between the consonants, and if it's preceded by any letter but *l*, break as *-ing.* If the letter before *-ling* is *b, c, d, f, g, k, p, t,* or *z*, break before this letter (but do not break a *ck* combination before *-ling*); otherwise break *-ing.*

If an *-able, -ary, -ful, -ize, -less, -ly, -ment,* or *-ness* is recognized, then the algorithm is applied again to the residual word (the word less the suffix we just found). Similarly, if we fail to find a suffix and the word ends in *s, e,* or *ed,* we try again with the *s, e,* or *ed* removed.

Having removed suffixes, we now try to remove potential prefixes, working on the word without all the suffixes just recognized. The following prefixes will be recognized:

> *be-, com-, con-, dis-, equi-, equiv-, ex-, hand-, horse-, hy-per-, im-, in-, in-ter, in-tro-, lex-i-, mac-ro-, math-e-, max-i-, min-i-, mul-ti-, non-, out-, over-, pseu-do-, quad-, semi-, some-, sub-, su-per-, there-, trans-, tri-, un-der,* and *un-.*

The intermediate hyphens aren't inserted unless the entire prefix is found (e.g., hy-per-bole vs. hyp-notic). There are also exceptions here:

Do you know where your bugs are?

This C programmer is finding his bugs the hard way...one at a time. That's why it's taking so long. But there's an easier way. Use

PC—Lint

PC—Lint* analyzes your C programs (one or many modules) and uncovers glitches, bugs, quirks, and inconsistencies. It will catch subtle errors before they catch you. PC—Lint resembles the Lint that runs on the UNIX* O.S., but with more features and some awareness of the 8086 environment.

- Full K&R C
- Supports Multiple Modules—finds inconsistencies between declarations and use of functions and data across a set of modules comprising a program.
- Compares function arguments with the associated parameters and complains if there is a mismatch or too many or too few arguments.
- User-modifiable library description files for most major compilers.
- All warning and information messages may be turned on and off globally or locally (via command line and comments) so that messages can be tailored to your programming style.
- All command line information can be furnished indirectly via file(s) to automate testing.
- Use it to check existing programs, programs about to be exported or imported, as a preliminary to compilation, or prior to scaling up to a larger memory model.
- All one pass with an integrated preprocessor so it's very fast.
- Has numerous flags to support a wide variety of C's, memory models, and programming styles.
- **Price: $139.00 MC, VISA**
  (Includes shipping and handling) PA residents add 6% sales tax. Outside USA add $10.00
- Runs under MS-DOS* 2.0 and up, with a minimum of 128Kb of memory. It will use all the memory available.

Trademarks: PC—Lint *(Gimpel Software)*, UNIX *AT&T)*, MS-DOS *(Microsoft)*).

GIMPEL SOFTWARE

3207 Hogarth Lane • Collegeville, PA 19426

(215) 584-4261

Be- must be followed by *c*, *h*, *s*, or *w*; *dis-* may not be followed by *h* or *y*; *equi-* may not be followed by *v*; *trans-* must be followed by *a*, *f*, *g*, *l*, or *m*; *tri-* must be followed by *a*, *f*, or *u*; *un-* may not be followed by *der* or *i*.

As with suffixes, the algorithm is applied repeatedly after *dis-*, *im-*, *in-*, *non-*, *over-*, or *un-* is recognized.

At this juncture, the word has been partitioned into three parts: prefix, root, and suffix. We now apply several consonant-pair rules to the root portion of the word. Here, vowels are *a*, *e*, *i*, *o*, *u*, and *y*, and consonants are all other single letters. Moreover, the pairs *ch*, *gh*, *ph*, *sh*, and *th* are treated as single consonants. If a vowel-consonant-consonant (VCC) pattern is found and both consonants are the same and neither consonant is an *l* or *s*, the break will be between the consonants. In the case of a *vowel-ll* or *vowel-ss* pattern, there will be a break only if the following letter isn't a vowel and the word doesn't end VCC*er* or VCC*ers*. If the pattern is *vowel-ck*, the break will be after the *ck*. If the pattern is *vowel-qu*, the break will be before the *qu*.

If a VCCV combination is found, the break will be between the consonants unless they are one of the following combinations:

bl, *br*, *cl*, *cr*, *chl*, *chr*, *dg*, *dr*, *fl*, *fr*, *ght*, *gl*, *gr*, *kn*, *lk*, *lq*, *nch*, *nk*, *nx*, *phr*, *pl*, *pr*, *rk*, *sp*, *sq*, *tch*, *tr*, *thr*, *wh*, *wl*, *wn*, or *wr*.

Finally, don't break between consonants if the word ends with VCC*er*, VCC*ers*, VCC*age*, VCC*ages*, VCC*est* and the two consonants are either *ft*, *ld*, *mp*, *nd*, *ng*, *ns*, *nt*, *rg*, *rm*, *rn*, *rt*, or *st*.

Once the word is hyphenated, Knuth goes back and fixes obvious mistakes. He takes back very short syllables and syllables ending with a silent *e*. I decided to use the exception dictionary to do this rather than adding more code to the algorithm proper. I should say that Knuth is not his usual lucid self when describing this part of the algorithm. A major factor in my not implementing the "short end" retention is that I can't, for the life of me, figure out what he was saying. If anybody can decipher this part of the algorithm description, please enlighten me.

Implementation

The code this month (Listing Two, page 26) supports the suffix removal, prefix removal, and consonant-pairs part of the algorithm. The Lattice compiler produces a 9063-byte object module (which seems a little bloated to me, though the code could be optimized a bit more if space is a problem). Hyphens are marked by setting the high bit of the character that will follow the hyphen (the first letter in the syllable). Only unhyphenated words composed of lower case letters will be processed. Among other things, this lets us avoid the problems associated with proper nouns, which shouldn't be hyphenated.

All global variables and all subroutines that don't need to be accessed from another module are declared static. The one externally accessible routine is the driver, **hyphen( )** (on line 484), which calls the various other processing routines.

The suffix/prefix removal routines use table-driven state machines for pattern recognition. As usual, there is a trade-off between space and execution time. The fastest state machines use a large two-dimensional array in which one axis is the current state and the other the current input character. The table itself holds the next state. As most of the states have only one legal transition, this method uses an unnecessarily large amount of memory. At the other extreme, we can keep a list associated with each state that contains all legal input characters and the state to go to when that character is encountered. This method usually takes less space, but we have to search the list every time we process a character so execution time isn't great when the lists are too long.

The tables used here are a compromise between these two methods. Each table is an array of pointers to

Prefixes:

0	Failure, return original pointer
1-68	Get another character and branch to next state as indicated in table.
69	Get a character, hyphenate ***p**, branch to next state using ***p** as the current character.
70	hyphenate ***(p–1)** and branch as above.
71-80	don't exist.
81	hyphenate ***(p–1)** and return **(p–1)**.
82	hyphenate ***p** and ***(p–1)**, return **p**.
83	hyphenate ***p** and ***(p–2)**, return **p**.
84	hyphenate ***p** and ***(p–3)**, return **p**.
85	hyphenate ***(--p)**, don't return.
86	hyphenate ***p**, dont't return.
87	hyphenate ***p** and return **p**.

Suffixes:

0	Failure, if the word ends in *s*, *e* or *ed* strip it off and try again, else return original string pointer.
1-57	Get another character and branch to next state as indicated in table.
58-79	Don't exist.
80	**p += 1**, hyphenate ***(p+1)**.
81	**p += 2**, hyphenate ***(p+1)**.
82	**p += 3**, hyphenate ***(p+1)**.
83	hyphenate ***(p+1)**.
84	Failure, return original pointer.
85	Doesn't exist.
86	process an -ing.
87	hyphenate ***(p+1)** and return **p**.
88	hyphenate ***(p+2)** and return **(p+1)**.
89	hyphenate ***(p+5)** and ***(p+1)**, return **p**.

In both tables, unspecified transitions go to state 0. Unmarked transitions always go to the indicated state, regardless of input character. **P** points at the character being processed at present.

Figure 1
States and Associated Actions

character arrays. The tables are indexed by current state. If there's only one legal transition out of a given state, the first byte of the associated array is 1, the next byte is the one legal input character, and the third byte is the state to go to when that character is found. All other input characters will cause a transition to state 0. The second type of array is used when there is more than one legal transition out of a state. In this case, the first byte of the array is 0 and the next 26 bytes are an array of next states indexed by current input character. The state tables begin on line 605 of the listing.

The state diagrams for these tables are shown in Figure 2 (page 24) and Figure 3 (page 25). In both machines, state 0 is a failure state. States with numbers greater than or equal to 80 are success states. These are the only states in which any sort of action is performed. The actions themselves are spelled out in Figure 1 (page 22). Note that the suffix recognition code goes through the word from back to front. Consequently, the state table for suffixes is a little weird looking (all the suffixes are backwards).

The table-processing routines start on lines 110 (suffix) and 221 (prefix). They are essentially the same, though different tables are used to do the processing. The subroutine **next**( ), when given the current state and input character, will return the next state. All possible action states are taken care of in the switches on lines 155 and 235.

Consonant-pair recognition is done by brute force rather than with a machine. (I tried to do a state machine, but it got too complicated). One of the places where the code size could be reduced is here. **Nextch**( ) (on line 318) gets the next input character and advances the string pointer past the character. If a consonant pair (*th*, *sh*, *ph*, *ch*, or *gh*) is found, both characters are skipped and an integer value (**#defined** on line 31 f.) is returned. The actual consonant-pair processing routine starts on line 393.

That's hyphenation. The algorithm works correctly with most of the words I've tried it on. My version successfully hyphenates su-per-califragi-lis-ticex-pialido-cious, just like Knuth's does. (Note that the algorithm doesn't insert every possible hyphen, but it doesn't put in any incorrect ones either.)

If anyone doesn't feel like typing in this month's program, machine readable copies of all source code from C Chest are now available on 5¼-inch IBM PC compatible disks from Software Engineering Consultants, P.O. Box 5679, Berkeley, CA 94705. The cost is $25 per disk (one disk is one month's column). In addition, my version of grep, the Unix generalized regular expression parser presented in *Doctor Dobb's Journal* #96, is still available for $35 from the same address.

Notes

[1] Donald E. Knuth, *T_EX and Metafont , New Directions in Typesetting* (Digital Press), pp.180–186. DDJ

(Listings begin on page 26)

Reader Ballot
Vote for your favorite feature/article.
Circle Reader Service **No. 191**.

EXTRA!... VALUE and PERFORMANCE with Relocatable Z-80 Macro Assembler
FROM MITEK

It's a real bargain! Here's why:

- Only $49.95 plus shipping
- 8080 to Z-80 Source Code Converter
- Generates Microsoft compatible REL files or INTEL compatible hex files
- Compatible with Digital Research macro assemblers MAC & RMAC
- Generates Ditigal Research compatible SYM files
- Full Zilog mnemonics
- INCLUDE and MACLIB files
- Conditional assembly
- Phase/dephase
- Separate data, program, common and absolute program spaces
- Customize the Macro Assembler to your requirements with installation program
- Cross-reference generation
- Supports Hitachi HD64180 additional instructions
- Z-80 Linker and Library Manager for Microsoft compatible REL files available as a total package with Macro Assembler for only $95.00 plus shipping; manual only is $15

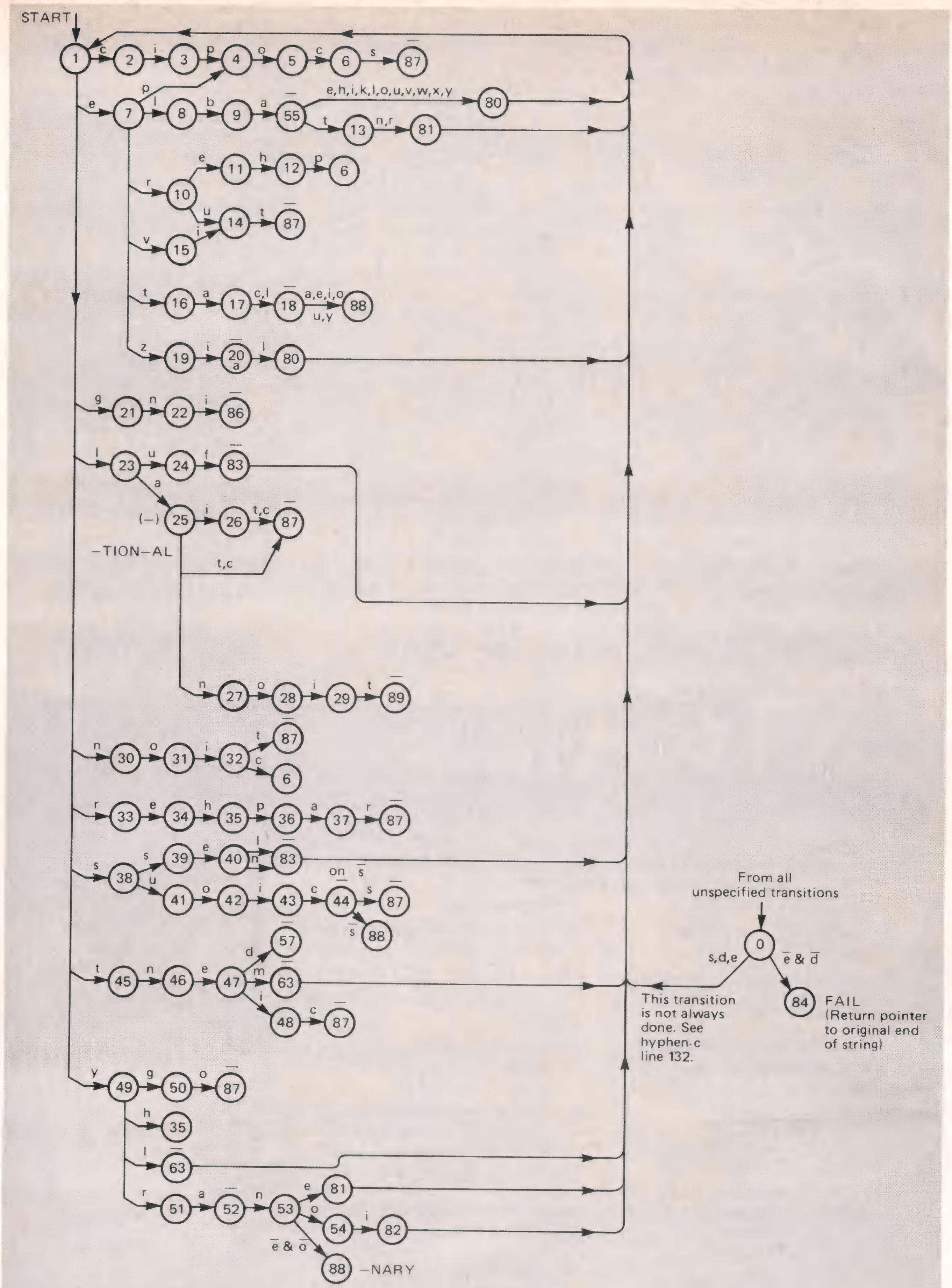
NEW for Turbo Pascal Users
Mitek's Relocatable Z-80 Macro Assembler will also:
- Generate Turbo Pascal in-line machine code include files
- Include files provide Turbo Pascal compatible machine code with assembly language mnemonics as comments

TO ORDER, CALL TOLL FREE: 1-800-367-5134, ext. 804
For information or technical assistance: 1-808-623-6361
Specify desired 5¼" or 8" format. Personal check, cashier's check, money order, VISA, MC, or COD welcomed. Include $5 for postage and handling.

Z-80 is a trademark of Zilog, Inc. MAC, RMAC, and ZSID are trademarks of Digital Research, Inc. Turbo Pascal is a trademark of Borland International, Inc.

MITEK
P.O. Box 2151
Honolulu, HI 96805

Circle **no. 66** on reader service card.



START
-TION-AL
(–)
-NARY
From all unspecified transitions
This transition is not always done. See hyphen.c line 132.
FAIL (Return pointer to original end of string)

Figure 1 SUFFIXES

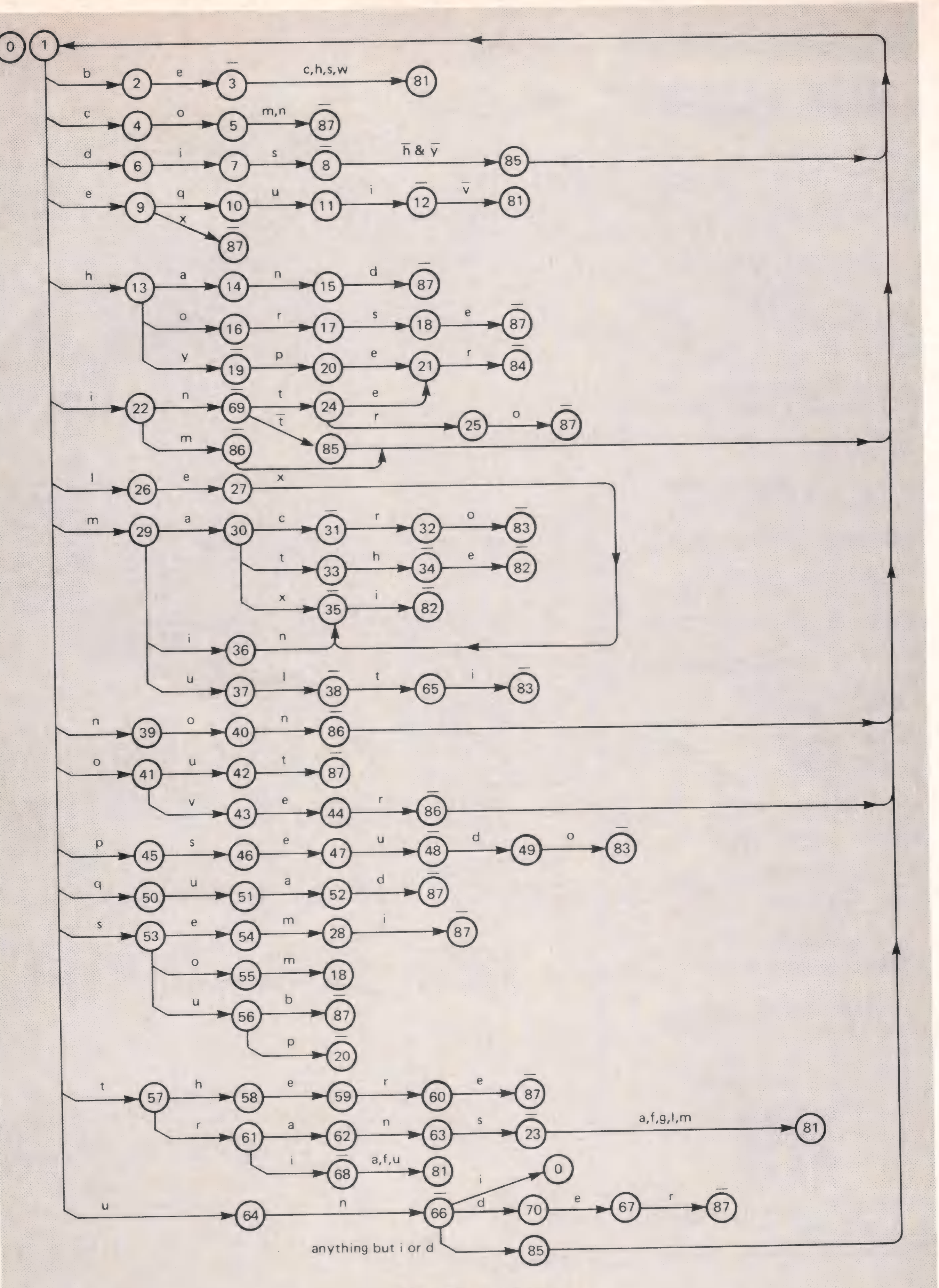


Figure 2 PREFIXES

NGS FORTH

A FAST FORTH, OPTIMIZED FOR THE IBM PERSONAL COMPUTER AND MS-DOS COMPATIBLES.

STANDARD FEATURES INCLUDE:

- 79 STANDARD
- DIRECT I/O ACCESS
- FULL ACCESS TO MS-DOS FILES AND FUNCTIONS
- ENVIRONMENT SAVE & LOAD
- MULTI-SEGMENTED FOR LARGE APPLICATIONS
- EXTENDED ADDRESSING
- MEMORY ALLOCATION CONFIGURABLE ON-LINE
- AUTO LOAD SCREEN BOOT
- LINE & SCREEN EDITORS
- DECOMPILER AND DEBUGGING AIDS
- 8088 ASSEMBLER
- GRAPHICS & SOUND
- NGS ENHANCEMENTS
- DETAILED MANUAL
- INEXPENSIVE UPGRADES
- NGS USER NEWSLETTER

A COMPLETE FORTH DEVELOPMENT SYSTEM.

PRICES START AT $70

NEW◆HP-150 & HP-110 VERSIONS AVAILABLE

NEXT GENERATION SYSTEMS
P.O.BOX 2987
SANTA CLARA, CA. 95055
(408) 241-5909

Circle no. 106 on reader service card.

C Chest **(Text begins on page 20)**

Listing One

```
+------------------------------------------------------------------+
|         Changes to ls.c to fix the root directory problem.        |
|                                                                    |
|         1) Add the subroutine isrootdir():                         |
+------------------------------------------------------------------+

isrootdir( name )
char *name;
{
        if( *name && name[1] == ':' )
                name += 2;

        return( (*name == '\\' || *name == '/') && !name[1] );
}

+------------------------------------------------------------------+
| 2) Modify the first if statement of the subroutine fixup_name()   |
|               (line 184 of the original listing) to read:         |
+------------------------------------------------------------------+

        if( !isrootdir(name)  &&  !find_first(name, ALL, regs) )
```

End Listing One

Listing Two

```
#include <stdio.h>
#define DEBUG

/*-------------------------------------------------------------------------*
 *      HYPHEN.C - An implementation of the Knuth hyphenation algorithm    *
 *                                                                          *
 *          Copyright (c) 1985, Allen I. Holub.  All rights reserved.       *
 *-------------------------------------------------------------------------*
 */

/* Various psuedo-subroutines. HYPHEN defines a bit to set when a hyphen
 * is inserted. HYPHENATE sets the bit, UNHYPHENATE clears it,
 * HAS_HYPHEN tests for it. The ER macro checks for an "er" at the end
 * of a word, it's used by the consonant pair checking routine.
 * Is consonant returns true if c is a consonant.
 */

#define HYPHEN           0x80
#define HYPHENATE(c)     ( (c) |=  HYPHEN )
#define UNHYPHENATE(c)   ( (c) &= ~HYPHEN )
#define HAS_HYPHEN(c)    ( (c) &   HYPHEN )

#define ER(p,end)        (*p == 'e' && *(p+1) == 'r' &&  (p+1) == end)
#define isconsonant(c)   ((c) && !isvowel(c))

/*      The dipthongs ch, gh, ph, sh and th are treated as single
 *      consonants. The subroutine nextch() will map these two
 *      characters into a single character as follows:
 */

#define CH ('z' + 1 )    /* {      0x7b  \173    */
#define GH ('z' + 2 )    /* |      0x7c  \174    */
#define PH ('z' + 3 )    /* }      0x7d  \175    */
#define SH ('z' + 4 )    /* ~      0x7e  \176    */
#define TH ('z' + 5 )    /* DEL    0x7f  \177    */

/*      S(x) is used to initialize the state tables. It turned out to
 *      be easier to use a define than a typedef. ATYPE is used store
 *      strings in something other than chars (useful if we're keeping
 *      attributes around). I haven't tried compiling with ATYPE set to
 *      anything except char so be careful if you do.
 */

#define S(x)  static char x[]
typedef char  ATYPE;

#define LATTICE 1           /* Compile for Lattice C compiler      */

/*-------------------------------------------------------------------------
 *      GLOBAL VARIABLES:
 */

static char **States;     /* Points to table for current state machine  */
extern char *Suffixes[]; /* Suffix state table (at end of this module) */
extern char *Prefixes[]; /* Prefix state table             "           */

#ifdef DEBUG
static int  Debug = 0;   /* True if debug diagnostics are to be printed */
#endif

/*-------------------------------------------------------------------------*/

#ifdef LATTICE

```

(Continued on page 28)

PERFORMANCE PACKAGE

Blaise Computing Inc. introduces the PERFORMANCE PACKAGE™ for Turbo Pascal programmers.

TURBO PASCAL

NEW!

Turbo ASYNCH™

With Turbo ASYNCH, you can be in constant touch with the world without ever leaving the console. Rapid transit at its best. Turbo ASYNCH is designed to let you incorporate asynchronous communication capabilities into your Turbo Pascal application programs, and it will drive any asynchronous device via the RS232 ports, like printers, plotters, modems or even other computers. Turbo ASYNCH is fast, accurate and lives up to its specs. Features include…

◆ Initialization of the COM ports allowing you to set all transmission options. ◆ Interrupt processing. ◆ Data transfer between circular queues and communications ports. ◆ Simultaneous buffered input and output to both COM ports. ◆ Transmission speeds up to 9600 Baud. ◆ Input and output queues as large as you wish. ◆ XON/XOFF protocol.

The underlying functions of Turbo ASYNCH are carefully crafted in assembler for efficiency, and drive the UART and programmable interrupt controller chips directly. These functions, installed as a runtime resident system, require just 3.2K bytes. The interface to the assembler routines is written in Turbo Pascal.

The Turbo Pascal PERFORMANCE PACKAGE™ is for the serious Turbo Pascal programmer who wants quality tools to develop applications. Every system comes with a comprehensive User Reference Manual, all source code and useful sample programs. They require an IBM PC or compatible, utilizing MS-DOS version 2.0 or later. There are no royalties for incorporating PERFORMANCE PACKAGE functions into your applications.

Turbo POWER TOOLS and Turbo ASYNCH sell for $99.95 each, and they may be ordered directly from **Blaise Computing, Inc. TO ORDER, call (415) 540-5441**

Turbo POWER TOOLS™

Turbo POWER TOOLS is a sleek new series of procedures designed specifically to complement Turbo Pascal on IBM and compatible computers. Every component in Turbo POWER TOOLS is precision engineered to give you fluid and responsive handling, with all the options you need packed into its clean lines. High performance and full instrumentation, including…

◆ Extensive string handling to complement the powerful Turbo Pascal functions. ◆ Screen support and window management, giving you fast direct access to the screen without using BIOS calls. ◆ Access to BIOS and DOS services, including DOS 3.0 and the IBM AT. ◆ Full program control by allowing you to execute any other program from within your Turbo Pascal application. ◆ Interrupt service routines written entirely in Turbo Pascal. Assembly code is not required even to service hardware interrupts like the keyboard or clock.

Using Turbo POWER TOOLS, you can now "filter" the keyboard or even DOS, and create your own "sidekickable" applications.

YES, send me the Best for the Best! Enclosed is ________ for
☐ Turbo ASYNCH ☐ Turbo POWER TOOLS. (CA residents add 6½% Sales Tax. All orders add $6.00 for shipping.)

Name: ____________ **Phone:** ____________
Shipping Address: ____________ **Zip:** ____________
City: ____________ **State:** ____________
VISA or MC #: ____________ **Exp. Date:** ____________

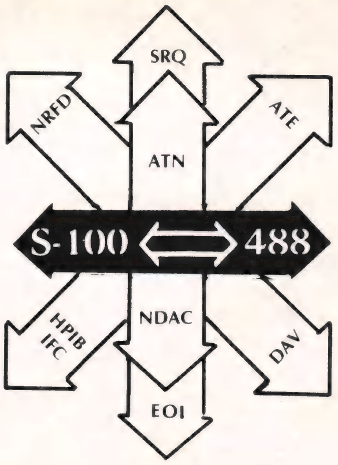
Turbo Pascal is a trademark of Borland International. Turbo POWER TOOLS, Turbo ASYNCH and PERFORMANCE PACKAGE are trademarks of Blaise Computing Inc. IBM is a registered trademark of International Business Machines Corporation. MS-DOS is a trademark of Microsoft Corporation.

BLAISE COMPUTING INC.

BLAISE *Watch us!*

◆ 2034 BLAKE STREET
◆ BERKELEY, CA 94704
◆ (415) 540-5441

Circle no. 8 on reader service card.

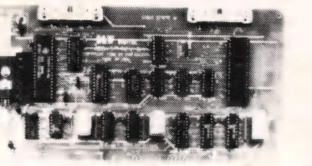


IEEE 488 TO S-100

INTERFACE

- Controls IEEE 488 (HPIB) Instruments with an S-100 computer
- Acts as controller or device
- Basic and assembly language drivers supplied
- Meets IEEE 696 specification
- Industrial quality burned in and tested up to 125K bytes/sec under software control 3 parallel ports (8255-5)
- $375

THE 488+3



DWD

D&W DIGITAL, INC.
20655 Hathaway Avenue
Hayward, California 94541
(415) 887-5711

Circle **no. 28** on reader service card.

C Chest

(Listing continued, text begins on page 20)

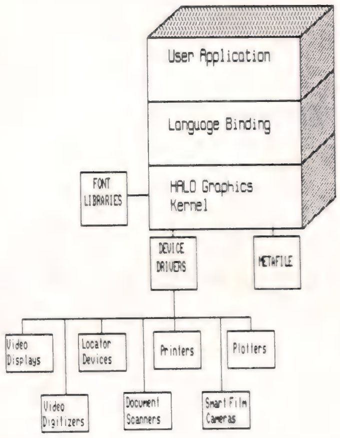
Listing Two

```
extern  char          *stpchr();
#define index(s,c)    stpchr(s,c) /* For some reason the unix index()
                                   * function is called stpchr() by
                                   * Lattice.
                                   */
#define register      static      /* This will give more efficient code
                                   * since Lattice doesn't support any
                                   * register variables. DON'T EXPLICITLY
                                   * INITIALIZE AUTOMATIC VARIABLES IF
                                   * THIS #DEFINE IS ACTIVE.
                                   */

#else   /*- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

char    *index(s,c)
char    *s;
{
        /*      Return a pointer to the left-most occurance of the
         *      character c in the string s or NULL if the character
         *      isn't found.
         */

        for(; *s ; s++ )
                if( *s == c )
                        return s;
        return NULL;
}

#endif

/*----------------------------------------------------------------------*/

isvowel(c)
ATYPE   c ;
{
        /* We check for a vowel in the order in which it is likely to
         * appear in English (ETAOIN SHRDLU). Return true if c is a vowel.
         */

        c &= 0x7f;
        return( c=='e' || c=='a' || c=='o' || c=='i' || c=='u' || c=='y');
}

/*----------------------------------------------------------------------*/

static  ATYPE   *suffix( beg, end )
ATYPE   *beg, *end ;
{
        /*      Split off any suffixes, using the "Suffixes" state table
         *      (defined below) to recognize them.
         */

        register char   *p;
        register int    state, c, c2, times ;

        state   = 1;
        times   = 0;
        States  = Suffixes;

        for( p = end; p >= beg ;)
        {
                state = next( state, c = *p-- & 0x7f  );

                if( !('a' <= c && c <= 'z') ) /* Words containing non-
                        return end;           /* lower-case characters *.
                                              /* won't be hyphenated.  */

                /* Test for state 0 with an if instead of in the switch.
                 * Lattice C will make the switch very inefficient if the
                 * range of case values is too high. State 0 is wierd;
                 * we use it to test for trailing e or ed on failure.
                 * The first time through we set p to end so we can check
                 * the end of the word. The third time through we abort.
                 * This means that we'll retry if the word ends -s -e -d
                 * -ed -de -dd -ee. The extra endings shouldn't matter.
                 */

                if( state == 0 )
                {
                        if( times == 0 )
                                p = end ;     /* Strip trailing e or d */

                        else if( times == 1 ) /* Strip trailing ed, de */
                                p = end-1 ;   /* ee, or dd             */
                        else
                                return end;

                        times++;
                }

                switch( state )
                {
```

(Continued on page 30)

HALO the market standard for graphics

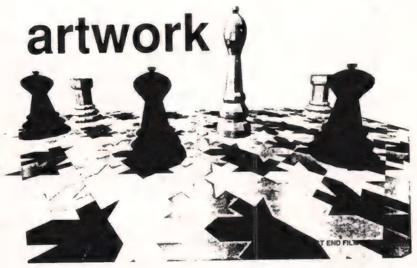
HALO provides software implementors with a complete graphics development environment.



Device Independent

Each copy of HALO comes with a full complement of display, input, printer, film recorder and plotter device drivers. Metafile creation and display are included. Media Cybernetics is continually developing new device drivers to meet the ever changing new hardware technologies.

HALO supports all IBM compatible MS/PC DOS systems including Texas Instruments Professional™, AT&T 6300™, IBM™ 3270-PC and of course the IBM PC, XT and AT. Over twenty graphics display boards manufactured by IBM and leading third party vendors are supported. A single device driver supports all display modes on each device. Image input from video digitizers and document scanners is supported. New device drivers include the IBM EGA and Professional and the AT&T Image Capture and Display boards.



This picture was created with "Artwork", a HALO-based application written by West End Film of Washington, DC.

De-Facto Standard

First released in 1982, HALO has an established base of over 40,000 end users and over 100 corporate clients. Numerous HALO-based CAD, solids modeling, presentation graphics, art packages, mapping and other vertical market software applications are commercially available.

As early as June 1984, PC World featured an article entitled, "HALO A new software library leads the way toward graphics standardization and portability." The July 9, 1985 issue of PC Week" featured users stories from both Rockwell International Corp., and Lawrence Livermore Labs on how they use HALO to save development time and money. Most recently, Mini-Micro Systems August 1985 issue stated, "Widely used, HALO has attained de facto-standard status. It's certainly the most widely used library." HALO has achieved this status because it provides a complete device independent graphics environment for software developers. Since the HALO interface rarely changes, compatibility with a new device is achieved simply by adding a new device driver.

By incorporating all the HALO device drivers into their applications, licensed commercial software developers solve distribution and compatibility problems.

This picture was created with "Design Board 3-D", a HALO-based application written by MEGA-CADD of Seattle, Washington.

Powerful Graphics Toolbox

HALO has over 170 functions. Standard functions include: point, line, arc, circle, ellipse, polygon, etc. Coloring control functions are available for color selection, palette management, dithering, and textures.

Advanced features encompass: image compression routines for reducing data storage requirements; both absolute and relative addressing plus individual cursors for graphics text and interaction; world and normalized coordinate systems for windowing, viewport and clipping; and a metafile reader and generator.

HALO's raster extension supports rubberbanding and non-destructive modes so that polygons and text may be interactively created and moved over a displayed image.

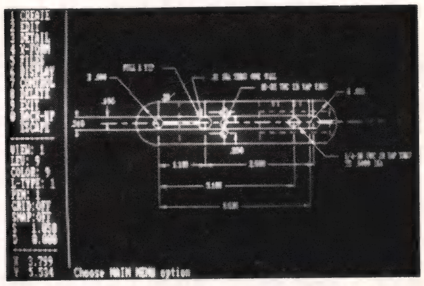
There are currently over twenty bit mapped and stroke text fonts available for HALO. The object oriented type faces provide user definition of line width, size, proportional width, angle, and interior fill style.

HALO's Virtual Rasterization Interface (VRI) frees users from the limited resolution of their graphics board by modeling the graphics display in memory and having the hardcopy produced in the maximum resolution of the output device.

HALO can be addressed as a linkable library or as a resident driver depending on the application.

Faster than VDI

In benchmark tests by independent companies evaluating HALO and GKS, HALO was determined to be three to ten times faster depending on the function tested. These benchmark results made HALO a requirement for many government agencies and large corporations. This gives HALO compatible hardware and software significant inroads into these agencies and corporations.



This picture was created with "CADKEY", a HALO-based application written by MICRO CONTROL SYSTEMS of Vernon, Connecticut.

Easy to Learn

HALO comes with a computer aided tutorial. This tutorial, entitled LEARNHALO, teaches users to become proficient graphics developers in less than a day by providing step-by-step instructions through graded examples.

HALO is written in Assembler and callable from the language you are already comfortable with: Interpretive and Compiled Basic, Professional Fortran, MS Fortran™, Pascal, Turbo Pascal™, MS C™, Mark Williams C™, Lattice C™, Aztec C™, Computer Innovations C86™, and Golden Common Lisp™.

Increase Market Value of Hardware

Graphics hardware manufactures receive distinct marketing advantages by having their products supported by HALO. HALO is the gateway to an extensive body of commercially available software.

Pricing Information

A single copy of HALO for one language costs $250.00. A second language binding ordered at the same time is an additional $150.00. Author, Corporate and Site licensing and OEM agreements are negotiated on a case by case basis.

To order, or receive more information on HALO and other Media Cybernetics, Inc., products contact:

media cybernetics, inc.
7050 Carroll Avenue
Takoma Park, MD 20912
(800) 446-HALO or (301) 270-0240
Telex #332014

HALO, LEARNHALO, Texas Instruments Professional, AT&T, IBM, MS Fortran, Microsoft C, Turbo Pascal, Mark Williams C, Lattice C, Aztec C, Computer Innovations C86, Golden Common Lisp and GKS VDI are all registered trademarks of Media Cybernetics Inc., International Business Machines, Texas Instruments Inc., AT&T, Microsoft Corp., Borland Intl., Mark Williams Co., Lattice Inc., Manx Software Systems, Computer Innovations Inc., Gold Hill Computers, and Graphics Software Systems Inc., respectively.

Circle **no. 67** on reader service card.

YOU NEED A GOOD LIBRARY

POWER PACKS™

COMPLETE SOURCES NO ROYALTIES

COMPREHENSIVE C Power Packs include over 1000 functions which provide an integrated environment for developing your applications efficiently. "This is a beautifully documented, incredibly comprehensive set of C Function Libraries."
— Dr. Dobb's Journal, July 1984

USEFUL "...can be used as an excellent learning tool for beginning C Programmers..."
— PC User's Group of Colorado, Jan. 1985

FLEXIBLE Most Compilers and all Memory Models supported.

RECOMMENDED "I have no hesitation in recommending it to any programmer interested in producing more applications code, using more of the PC capabilities, in much less time." — Microsystems, Oct. 1984

- **PACK 1: Building Blocks I** $149 DOS, Keyboard, File, Printer, Video, Async
- **PACK 2: Database** $399 B-Tree, Virtual Memory, Lists, Variable Records
- **PACK 3: Communications** $149 Smartmodem™, Xon/Xoff, X-Modem, Modem-7
- **PACK 4: Building Blocks II** $149 Dates, Textwindows, Menus, Data Compression, Graphics
- **PACK 5: Mathematics I** $99 Log, Trig, Random, Std Deviation
- **PACK 6: Utilities I (EXE files)** $99 Arc, Diff, Replace, Scan, Wipe

Master Card/Visa, $7 Shipping, Mass. Sales Tax 5%

ASK FOR FREE DEMO DISKETTE

NOVUM ORGANUM INC. SOFTWARE HORIZONS INC.

165 Bedford St., Burlington, MA 01803
(617) 273-4711

Circle no. **140** on reader service card.

C Chest

(Listing continued, text begins on page 20)

Listing Two

```
                case 86:
                        /*
                         * Process -ing. This could have been done with
                         * more states but it seemed easier to do it here.
                         */

                        c =     *p   & 0x7f ;
                        c2 = *(p-1) & 0x7f ;

                        if( p - beg < 3 )
                        {
                                return end;     /* FAIL */
                        }
                        else if(      ( c == c2 ) && ( !isvowel(c) )
                                   && ( c != 'f') && ( c != 's'     )
                                   && ( c != 'l') && ( c != 'z'     ) )
                        {
                                --p;
                        }
                        else if( c == 'l'  &&  index("bcdfghkptz", c2)  )
                        {
                               p -= (c2 == 'k' && (*(p-2) & 0x7f) == 'c')
                                                                 ? 1 : 2 ;
                        }

                        /* fall through to state 87 */

                case 87:
                        HYPHENATE( *(p+1) );
                        return p;

                case 88:
                        HYPHENATE( *(p+2) );
                        return( p + 1 );

                case 89:
                        HYPHENATE( *(p+5) );
                        HYPHENATE( *(p+1) );
                        return p;

                case 82: p++;     /* p += 3 */
                case 81: p++;     /* p += 2 */
                case 80: p++;     /* p += 1 */
                case 83:
                        end = p ;
                        HYPHENATE( *(p + 1) );
                        state = 1;
                        break;

                case 84:                                /* FAILURE STATE */
                        return end;
                }
        }
#ifdef DEBUG
        if( Debug )
                printf("\n");
#endif

        return end;
}

/*------------------------------------------------------------------------*/

static  ATYPE   *prefix( beg, end )
ATYPE   *beg, *end;
{
        /*      Split off any prefixes, using the "Prefixes" state table
         */

        register ATYPE  *p;
        register int    state, c ;

        state  = 1;
        States = Prefixes;

        for( p = beg ; p < end ;)
        {
                switch( state = next(state, c = *p++ & 0x7f) )
                {
                case 82: HYPHENATE( *p ); HYPHENATE( *(p-1) ) ; return p;
                case 83: HYPHENATE( *p ); HYPHENATE( *(p-2) ) ; return
                case 84: HYPHENATE( *p ); HYPHENATE( *(p-3) ) ; return p

                case 81: --p;
                case 87: HYPHENATE( *p );
                         return p;

                case 85: --p;
                case 86: beg   = p;
                         state = 1;
                case 69: HYPHENATE( *p );
```

```
                        break;

                case 70: HYPHENATE( *(p-1) );
                        break;

                /*      The following states don't actually exist.
                 *      Putting them in the table will cause Lattice to
                 *      compile the switch in a more efficient manner.
                 *      (see p 4-33 of the 1985 vintage manual).
                 */

                case 71: case 72: case 73: case 74: case 75:
                case 76: case 77: case 78: case 79:
                        break;
                }
        }

#ifdef DEBUG
        if( Debug )
                printf("\n");
#endif

        return beg;     /* Failure */
}

/*-----------------------------------------------------------------
 *      This table is used to find exceptions to the VCCV rule. It is
 *      indexed by the first of the two consonents & contains pointers
 *      to strings, any character of which will form an exception if
 *      it's the second consonant.
 */

static char     *vccv_except[] =
{
        /* a  */        ""      ,
        /* b  */        "lr"    ,
        /* c  */        "lr"    ,
        /* d  */        "gr"    ,
        /* e  */        ""      ,
        /* f  */        "lr"    ,
        /* g  */        "lr"    ,
        /* h  */        ""      ,
        /* i  */        ""      ,
        /* j  */        ""      ,
        /* k  */        "n"     ,
        /* l  */        "kq"    ,
        /* m  */        ""      ,
        /* n  */        "{kx"   , /* CH, k, x */
        /* o  */        ""      ,
        /* p  */        "lr"    ,
        /* q  */        ""      ,
        /* r  */        "k"     ,
        /* s  */        "pq"    ,
        /* t  */        "{r"    , /* CH, r    */
        /* u  */        ""      ,
        /* v  */        ""      ,
        /* w  */        "hlnr"  ,
        /* x  */        ""      ,
        /* y  */        ""      ,
        /* z  */        ""      ,
        /* CH */        "lr"    ,
        /* GH */        "t"     ,
        /* PH */        "r"     ,
        /* SH */        ""      ,
        /* TH */        "r"
};

/*-----------------------------------------------------------------*/

static  ATYPE   nextch( pp, endp )
ATYPE   **pp, *endp;
{
        /*  Advance *pp (one for a vowel or consonant, 2 for a dipthong)
         *  and return the character we've just skipped. Dipthongs are
         *  mapped to the single characters: TH, SH, PH, CH or GH.
         *  0 is returned once *pp reaches or passes endp.
         */

        register ATYPE  rval, *p;

        if( (p = *pp)  > endp )
                return (ATYPE) 0;

        rval = *p++ & 0x7f ;

        if( ( *p & 0x7f )  ==  'h' )
        {
                switch( rval )
                {
                case 't':       rval = TH;      p++;    break;
                case 's':       rval = SH;      p++;    break;
                case 'p':       rval = PH;      p++;    break;
                case 'c':       rval = CH;      p++;    break;
                case 'g':       rval = GH;      p++;    break;
                }
```

(Continued on next page)

LISP

FOR THE IBM PERSONAL COMPUTER.

THE PREMIER LANGUAGE OF ARTIFICIAL INTELLIGENCE FOR YOUR IBM PC.

- **DATA TYPES**
  Lists and Symbols
  Unlimited Precision Integers
  Floating Point Numbers
  Character Strings
  Multidimensional Arrays
  Files
  Machine Language Code
- **MEMORY MANAGEMENT**
  Full Memory Space Supported
  Dynamic Allocation
  Compacting Garbage Collector
- **FUNCTION TYPES**
  EXPR/FEXPR/MACRO
  Machine Language Primitives
  Over 190 Primitive Functions
- **IO SUPPORT**
  Multiple Display Windows
  Cursor Control
  All Function Keys Supported
  Read and Splice Macros
  Disk Files
- **POWERFUL ERROR RECOVERY**
- **8087 SUPPORT**
- **COLOR GRAPHICS**
- **LISP LIBRARY**
  Structured Programming Macros
  Editor and Formatter
  Package Support
  Debugging Functions
  .OBJ File Loader
- **RUNS UNDER PC-DOS 1.1 or 2.0**

IQLISP

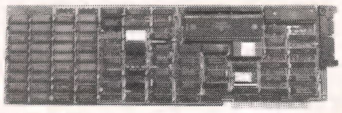
5¼" Diskette
and Manual ________ $175.00

∫q *Integral Quality*

P.O. Box 31970
Seattle, Washington 98103-0070
(206) 527-2918
Washington State residents add sales tax.
VISA and MASTERCARD accepted.
Shipping included for prepaid orders.

Circle no. **56** on reader service card.

68000
CO-PROCESSING
For
IBM PC, PC/XT
and
COMPATIBLE
SYSTEMS



Now you can add the MOTOROLA 68000 16/32 Bit Processor to your PC via use of the Pro 68 Advanced Technology Co-Processor. Enjoy all of the performance benefits of the 68000 processor without sacrificing your current PC system. Consider these impressive standard features of Pro 68:

- High Speed MOTOROLA 68000 micro processor
- 10Mhz no wait state design (3 times faster than the IBM PC/AT)
- True 16/32 bit technology
- For use on IBM PC, PC/XT or compatible systems
- On board 16 bit parity checked memory, 256K to 1024K
- Two serial I/O ports for multi user interface
- Provisions for the high speed NS32081 math processor
- High speed proprietary dual port host bus interface
- Parallel or array processing via multi processor architecture
- MS/PC DOS RAM disk driver program
- Choice of two popular integrated 16/32 bit operating systems:
  - CPM68K from Digital Research Inc.
    - Full suite of development tools
    - "C" compiler with floats and UNIX I/O library
    - Many third party compatible languages and applications
  - OS9/6800 from MICROWARE Corporation
    - UNIX look alike with multi user/multi tasking, shell, hierarchical disk directory, record and file lock, pipes and filters
    - Full suite of development tools
    - UNIX V compatible "C" compiler
    - Optional languages include BASIC, ISO PASCAL, FORTRAN 77.

Pricing from $1195 includes Pro68 with 256K, OS, and MS/PC DOS RAM disk driver. HSC also manufactures and markets a full line of co-processors and RAM disks for use on Z80 based systems.

DISTRIBUTORS:

Australia-Computer Transition Systems	...03-537-2768
Great Britain-System Science	...01-248-;062
West Germany-DSC International	...089-723-1125
Canada Remote Systems	...416-239-2835

Dealer, Distributor and OEM inquiries invited.

Hallock Systems Co., Inc.
267 North Main Street
Herkimer, NY 13350
(315) 866-7125

Circle no. **26** on reader service card.

C Chest

(Listing continued, text begins on page 20)

Listing Two

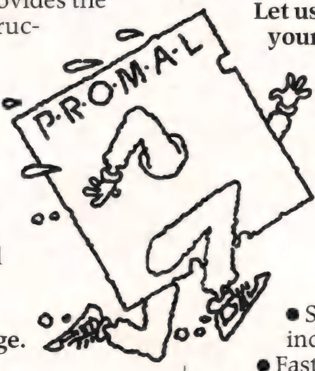
```
        }

#ifdef DEBUG
        if( Debug )
                printf( "nextch(), got <%c>\n", rval );
#endif

        *pp = p;
        return rval;
}


/*---------------------------------------------------------------------*/

static  iswierd( x, y, p )
ATYPE   *p;
{
        /* Return true if the string pointed to by p ends in:
         *      XYer XYers XYage XYages XYest
         *
         * where XY is one of the pairs:
         *      ft ld mp nd ng ns nt rg rm rn rt st
         */

        register int    c1, c2, c3;

        c1 = *p++ ;
        c2 = *p++ ;
        c3 = *p   ;

        return(
                (
                        ( c1 == 'e' && c2 == 'r'               )
                    ||  ( c1 == 'a' && c2 == 'g' &&  c3 == 'e' )
                    ||  ( c1 == 'e' && c2 == 's' &&  c3 == 't' )
                )
                  &&
                (       ( x == 'f'  && y == 't' )
                    ||  ( x == 'l'  && y == 'd' )
                    ||  ( x == 'm'  && y == 'p' )
                    ||  ( x == 's'  && y == 't' )
                    ||  ( x == 'n'  && index( "dgst", y) )
                    ||  ( x == 'r'  && index( "gmnt", y) )
                )
        );
}

/*--------------------------------------------------------------------*/

static  consonants( beg, end )
ATYPE   *beg, *end;
{
        /*      Hyphenate consonant pairs.
         *      Look for a VCC pattern: use the state machine:
         *
         *               V          C        C
         * Start ----> 1 -----> 2 -----> 3 -----> 4
         *         ^   |C   ^   | V      | V      | (no fetch )
         *         |   |    |   v        |        |
         *         |<--+    +------<-----+        |
         *         |                              |
         *         +------------------------------+
         *
         *    The machine is implemente explicitly (with whiles and ifs
         *    ifs and such) rather than with a table driven engine.
         *    A new character is fetched in states 1, 2, & 3 (but not 4).
         *    When you exit reach state 4 c1 and c2 will hold the two
         *    consonants, cp will point to the beginning of the second
         *    consonant, beg will point just past the second consonant.
         */
        register ATYPE  c1, c2, *cp;
        register char   *p;

        while( 1 )
        {
                state1:

                do {
                        c2 = nextch( &beg, end );
                } while( isconsonant(c2) );

                do {
                        for( c1 = c2; isvowel(c1) ; )
                        {
                                cp = beg;
                                c1 = nextch( &beg, end );
                        }


                        if(c1 == 'q' && *beg == 'u')     /*         Vqu */
                        {
                              HYPHENATE( *cp );          /*        V-qu */
```

(Continued on page 34)

Now there is an even better structured, compiled programming environment than PROMAL. Introducing PROMAL 2.0 for the IBM PC, the Apple II, and the Commodore 64.

Until now, the best next language for the serious programmer was PROMAL™ Now, it's the new PROMAL—PROMAL 2.0.

PROMAL 2.0 provides the same sophisticated structured programming environment, the same fast one-pass compiler, the same speed of execution, the same powerful commands of the earlier release —plus a host of useful new features.



Not just a language. A complete programming environment.

PROMAL—the PROgrammer's Micro Application Language—provides you with a complete programming environment, including a structured, high level language, a powerful program Editor, and a compiler that quickly turns your source code into compact, rapidly executing object code. Plus a library of integrated machine-language subroutines for frequently used tasks. And for the Apple II and the Commodore 64, PROMAL provides a DOS-like operating system Executive.

PROMAL 2.0—Even more of a good thing.

In addition to all of the features that have made PROMAL users declare it "the best language I've ever used," PROMAL 2.0 provides:

- Overlays that can be compiled separately for modular programming.
- Program size greater than 64k. (IBM PC only).
- True machine-to-machine portability.
- True 808X object code for the IBM PC.

Let us prove that PROMAL is your best next language!

Buy PROMAL 2.0 and try it for 15 days. If you don't believe it's your best next language, just return it for a full refund.

PROMAL Features

- Structured language with indentation.
- Fast, one-pass compiler.
- Simplified syntax.
- No line numbers.
- Multi-dimensional arrays, strings and pointers.
- Long variable names.
- Global, Local variables.
- Byte, Word, Integer & Real types.
- Decimal or Hex numbers.
- Functions and procedures with passed arguments.
- Built-in I/O library.
- Control Statements: IF-ELSE, IF, WHILE, FOR, CHOOSE, BREAK, REPEAT, INCLUDE, etc.
- Compiler I/O from/to disk or memory.

Executive*

Command oriented with line editing.
Allows multiple user programs in memory at once.
Function key redefinition.
Program abort or pause.
22 resident system commands.
Unlimited user-defined commands.
Prior command recall/edit.
I/O redirection to disk or printer.
Batch jobs.

Editor

Full-screen, cursor driven.
Function key controlled.
Line insert, delete, search.
String search and replace.
Block copy/move/delete/read/write.
Auto indent, undent support.
Edit after error.

Library

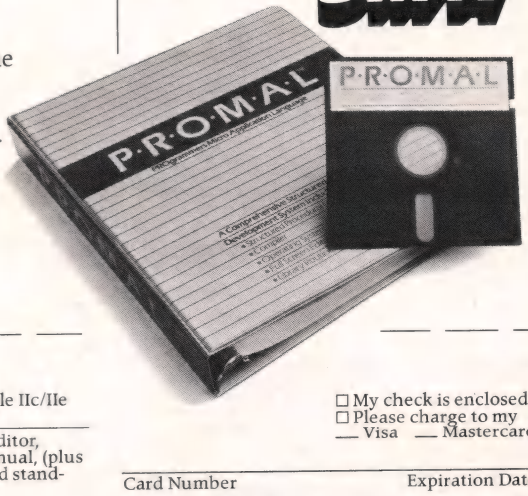
50 machine language commands.
Memory resident.
Call by name with arguments.
Formatted real output, string operations and much more.

*Apple II and Commodore 64 only. Requires one disk drive and 80-column card for Apple (IIe, IIc only).

1-800-762-7874
In NC: 919-878-3600

Systems Management Associates
3325 Executive Drive, Dept. D–1
Raleigh, North Carolina 27609

SMA



NEW for PROMAL The Graphics Toolbox*

Twenty fast subroutines for creating sophisticated, high-resolution graphics, including windows, clipping, scaling, and text-on graphics using scaled, rotated, user-defined fonts. $29.95.

*Available for the Apple II and the Commodore 64

Order Form

My system is (check one)
☐ IBM PC/100% compatibles ☐ Apple IIc/IIe
☐ Commodore 64/128

☐ Developer's Version—Compiler, Editor, Library, Demo disk, 280-page manual, (plus Executive for Apple and C-64) and stand-alone program generation. **$99.95 + 5.00 s/h.**

☐ End-User System for Apple II and Commodore 64—all features of Developer's version except stand-alone program generation. **$49.95 + 5.00 s/h.**

☐ Demo System—32-page "Meet PROMAL" manual and demonstration disk. **$10.00 + 2.50 s/h.**

☐ Graphics Tool Box for PROMAL—Available for Apple and C-64 only. **$29.95 + 2.50 s/h.**

☐ My check is enclosed.
☐ Please charge to my
___ Visa ___ Mastercard

Card Number ________ Expiration Date ________

Signature ________

Name ________

Address ________

City, State, Zip ________

NC residents add 4-1/2% sales tax.
Foreign orders add $15.00 additional s/h.

Circle no. **104** on reader service card.

FORTH

FREEDOM OF CHOICE

79 STANDARD

fig MODEL

IBM® TRS-80 CP/M

SOTA Computing Systems Limited lets you choose between either the versatile figFORTH model or the popular 79 Standard. Each version is available for a number of popular computer systems including the IBM PC, XT and AT (or compatibles), the TRS-80 Model I, III and 4/4P, or any computer system running CP/M (version 2.x) or CP/M Plus (version 3.x). What's more, SOTA doesn't require you to enter into any awkward or expensive royalty or licensing arrangements. As long as your applications programs do not offer the end user access to the basic FORTH system, you are free to make as many copies of the compiled FORTH system as you please and distribute them as you wish. FORTH from SOTA is the FORTH of choice for both the novice and experienced programmer. Make it your choice now! Order your copy today.

When you order from SOTA, both the fig model and 79 standard come complete with the following extra features at no additional charge:

▪ full featured string handling ▪ assembler ▪ screen editor ▪ floating point ▪ double word extension set ▪ relocating loader ▪ beginner's tutorial ▪ comprehensive programmer's guide ▪ exhaustive reference manual ▪ unparalleled ▪ technical support ▪ source listings ▪ ▪ unbeatable price ▪

ORDER FORM

GENTLEMEN: Rush me my order!

TOTAL US funds

☐ Enclosed is my ☐ check ☐ money-order
☐ Please bill my ☐ VISA ☐ MasterCard
for $89.95

Please send me ☐ 79 Standard FORTH ☐ figFORTH model for the
☐ IBM PC ☐ XT ☐ AT (and compatibles)
☐ TRS-80 Model I ☐ Model III ☐ Model 4 ☐ Model 4P
☐ CP/M Version 2.x ☐ CP/M Plus (Version 3.x)
For CP/M versions please note 5 1/4" formats only and please specify computer type

NAME:
STREET:
CITY/TOWN:
STATE: ZIP:
CARD TYPE: EXPIRY:
CARD NO:
SIGNATURE:

ORDER TODAY

213-1080 Broughton Street
Vancouver, British Columbia
Canada • V6G 2A8

MasterCard • Order by Mail or Phone (604) 688-5009 • VISA

State-of-the-Art since 1981

SOTA

Computing Systems Limited

IBM, TRS-80 and CP/M are registered trademarks of International Business Machine Corporation, Radio Shack and Digital Research respectively.

Circle **no. 100** on reader service card.

C Chest

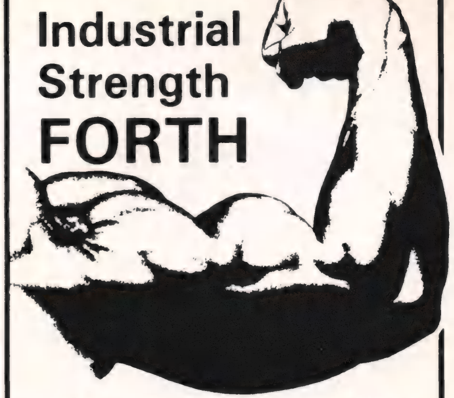
(Listing continued, text begins on page 20)

Listing Two

```
                                nextch( &beg, end );     /*  skip the u */
                                goto statel;
                        }

                        cp = beg ;
                        c2 = nextch(&beg, end) ;

                } while( isvowel(c2) ) ;

                if( !cl || !c2 )
                        break;

                /*      At this point we have found a Vowel-Cons-Cons
                 *      sequence. Cl and c2 will hold the left and right
                 *      consonant respectively. Beg will point at the
                 *      character following the second consonant. Cp
                 *      will point at the second consant.
                 */

                if( cl == 'c' &&  c2 == 'k' )           /*         Vck  */
                {
                        if( *beg )
                                HYPHENATE( *beg );      /*         Vck- *
                }
                else if( cl == c2 )
                {
                        if( (cl != 'l'  &&  cl != 's')  ||
                                (isvowel(*beg) && !ER(beg,end)) )
                                HYPHENATE( *cp );
                }

                else if( isvowel( *beg ) )              /*         VCCV */
                {
                        if( ! iswierd(cl, c2, beg) )
                        {
                                if( !((p = vccv_except[ (int)cl - 'a' ])
                                                       && index(p, c2) ))
                                       HYPHENATE( *cp );
                        }
                }
        }

        if( HAS_HYPHEN(*end) )
                UNHYPHENATE( *end );
}

/*----------------------------------------------------------------------*/

hyphen( beg, end )
ATYPE   *beg, *end;
{
        /*      Hyphenate the word deliniated by beg and end: First
         *      strip suffixes, then strip a trailing s e or ed,
         *      then strip prefixes. Only words with more than four
         *      letters and which consist of lower case letters only
         *      (no NULLs) will be hyphenated. 0 is returned if no
         *      attempt was made to hyphenate the word, one otherwise.
         *      If the word has already been hyphenated, 1 is returned
         *      but the word is not modified.
         */

        register ATYPE  *prefixp, *suffixp;

        if( end-beg <= 4 )
                return 0;

        for( prefixp = beg; prefixp <= end ; prefixp++ )
        {
                if( HAS_HYPHEN(*prefixp) )
                        return 1;

                if( !islower(*prefixp & 0x7f) )
                        return 0;
        }


        suffixp = suffix(beg,end);      /* Hyphenate and remove anv     */
                                        /*      suffixes
        if( suffixp == end )
        {
                /*      If root has a trailing s e or ed remove it
                 *      before applying any other rules
                 */

                if(*end == 's' || *end =='e')
                        suffixp = end - 1 ;

                else if( *(end-1) == 'e'  &&  *end == 'd' )
                        suffixp = end - 2 ;
        }

        prefixp = prefix( beg, suffixp);  /* Hyphenate and remove any */
                                          /* prefixes                 */
#ifdef DEBUG
        /*-------------------------------------------------------*/
        /*  Print the word in the form:  prefixes/root/suffixes */
        /*                                                      */
        bprint( beg, prefixp-1  );      /* Print prefixes       */
        putchar('/');
        bprint( prefixp, suffixp );     /* Print middle         */
        putchar('/');
        bprint( suffixp+1, end   );     /* Print suffixes       */
        /*------------------------------------------------------*/
#endif

        if( (suffixp - prefixp)  >=  3 )
        {
                /* Apply the consonant pairs rules only if the word
                 * has at least 4 letters in it (after prefixes
                 * and suffixes have been removed
                 */

                consonants( prefixp, suffixp );
        }

        return 1;
}

/*-----------------------------------------------------------------------*/

static  next( cur_state, cur_char )
ATYPE   cur_char  ;
int     cur_state ;
{
        /*      Given the current state and the current input character
         *      return the next state. The global variable States must
         *      point at the correct state table (Suffixes or Prefixes).
         */

        char     *p = States[ cur_state ] ;

#ifndef DEBUG

        if( *p )
                return (int)( (cur_char == p[1]) ? p[2] : 0 );
        else
                return (int)( p[(cur_char - 'a') + 1] );
#else

        int      rval;
        if( Debug )
                printf("%s: Current state = %d, Input char = <%c>, ",
                                States == Prefixes ? "PREFIX" : "SUFFIX",
                                cur_state, cur_char );

        if( *p )
                rval = (int)( (cur_char == p[1]) ? p[2] : 0 );
        else
                rval = (int)( p[(cur_char - 'a') + 1] );

        if( Debug )
                printf("Next state = %d\n", rval );

        return( rval );
#endif
}

/*-----------------------------------------------------------------------
 * State machine tables:
 *
 * There are two types of states. If the first entry is 0 then the state
 * is an array holding the next state, indexed by the current input
 * character. If the first entry is 1 then all transitions but one are to
 * state 0, the next character is the one exception and the transition
 * state follows.
 *
 * -----------------------------------------------------------------------
 */

S(s0 )={ 0, 84,84,84,1,1,84,84,84,84,84,84,84,84,84,84,84,84,84,1,84,
                                                     84,84,84,84,84,84 };

S(s1 )={ 0, 0,0,2,0,7,0,21,0,0,0,0,23,0,30,0,0,0,33,38,45,0,0,0,0,49,0};
S(s2 )={ 1, 'i', 3  };
S(s3 )={ 1, 'p', 4  };
S(s4 )={ 1, 'o', 5  };
S(s5 )={ 1, 'c', 6  };
S(s6 )={ 1, 's', 87 };
S(s7 )={ 0, 0,0,0,0,0,0,0,0,0,0,0,8,0,0,0,4,0,10,0,16,0,15,0,0,0,19 };
S(s8 )={ 1, 'b', 9  };
S(s9 )={ 1, 'a', 55 };
S(s10)={ 0, 0,0,0,0,11,0,0,0,0,0,0,0,0,0,0,0,0,0,0,14,0,0,0,0,0 };
S(s11)={ 1, 'h', 12 };
S(s12)={ 1, 'p', 6  };
S(s13)={ 0, 0,0,0,0,0,0,0,0,0,0,0,0,81,0,0,0,81,0,0,0,0,0,0,0,0 };
S(s14)={ 1, 't', 87 };
S(s15)={ 1, 'i', 14 };
S(s16)={ 1, 'a', 17 };
S(s17)={ 0, 0,0,18,0,0,0,0,0,0,0,18,0,0,0,0,0,0,0,0,0,0,0,0,0,0 };
S(s18)={ 0, 88,0,0,0,88,0,0,0,88,0,0,0,0,0,88,0,0,0,0,0,88,0,0,0,88,0};
S(s19)={ 1, 'i', 20 };
S(s20)={ 1, 'l', 80 };
S(s21)={ 1, 'n', 22 };
S(s22)={ 1, 'i', 86 };
S(s23)={ 0, 25,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,24,0,0,0,0,0 };
```

(Continued on next page)

Mulituser/Multitasking for 8080, Z80, 8086



TaskFORTH™

The First Professional Quality Full Feature FORTH System at a micro price*

LOADS OF TIME SAVING PROFESSIONAL FEATURES:

- ★ Unlimited number of tasks
- ★ Multiple thread dictionary, superfast compilation
- ★ Novice Programmer Protection Package ™
- ★ Diagnostic Tools, quick and simple debugging
- ★ *Starting FORTH, FORTH-79, FORTH-83* compatible
- ★ Screen and serial editor, easy program generation
- ★ Hierarchical file system with data base management

* Starter package $250. Full package $395. Single user and commercial licenses available.

If you are an experienced FORTH programmer, this is the one you have been waiting for! If you are a beginning FORTH programmer, this will get you started right, and quickly too!

Available on 8" or 5¼" disk in various formats under CP/M 2.2 or greater and 5¼" MS-DOS

FULLY WARRANTIED, DOCUMENTED AND SUPPORTED

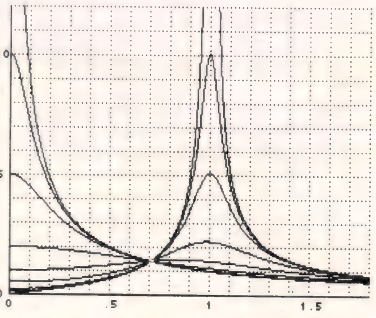
VISA DEALER INQUIRIES INVITED MasterCard

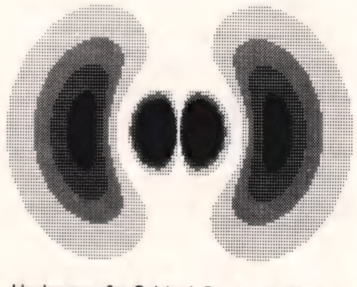
Shaw Laboratories, Ltd.
24301 Southland Drive, # 216
Hayward, California 94545
(415) 276-5953

Circle no. **124** on reader service card.

isys forth

for the Apple®] [


Parallel Resonance with Damping
BASIC 213 sec ISYS FORTH 20 sec


Hydrogen 3p Orbital Cross-section
BASIC 492 sec ISYS FORTH 39 sec

ISYS FORTH is a FORTH-83 compiler designed especially for scientific and engineering applications.

- **FASTEST** — Sieve benchmark, 3.3 seconds/pass. Compiles to machine language.
- **FLOATING POINT WITH TRANSCENDENTALS,** single and double precision.
- **16-BIT 65802 SUPPORT** for execution speeds 50-100% faster than the above.
- **FIXED POINT THEORY AND EXAMPLES** (a 30-page chapter). D*/ for double precision fixed point.
- **TURTLE AND CARTESIAN GRAPHICS** with 70-column character set and double hires support.
- **MACRO ASSEMBLER.**
- **DOS 3.3 FILE INTERFACE.**
- **FULL-SCREEN EDITOR.**
- **144-PAGE MANUAL WITH TUTORIAL.**
- **PRICE: $99, NO EXTRA CHARGES.**

ILLYES SYSTEMS
PO Box 2516, Sta A
Champaign, IL 61820
Phone: 217/359-6039

For any Apple] [model, 48K or larger.
Apple is a registered trademark of Apple Computer.

Circle **no. 46** on reader service card.

C Chest (Listing continued, text begins on page 20)

Listing Two

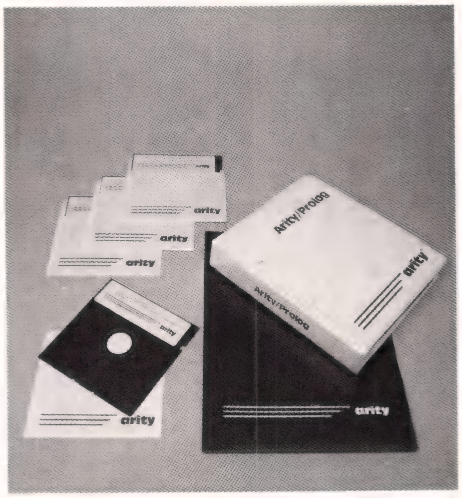
```
S(s24)={ 1, 'f', 83 };
S(s25)={ 0, 0,0,87,0,0,0,0,0,26,0,0,0,0,27,0,0,0,0,0,87,0,0,0,0,0,0 };
S(s26)={ 0, 0,0,87,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,87,0,0,0,0,0,0 };
S(s27)={ 1, 'o', 28 };
S(s28)={ 1, 'i', 29 };
S(s29)={ 1, 't', 89 };
S(s30)={ 1, 'o', 31 };
S(s31)={ 1, 'i', 32 };
S(s32)={ 0, 0,0,6,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,87,0,0,0,0,0,0 };
S(s33)={ 1, 'e', 34 };
S(s34)={ 1, 'h', 35 };
S(s35)={ 1, 'p', 36 };
S(s36)={ 1, 'a', 37 };
S(s37)={ 1, 'r', 87 };
S(s38)={ 0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,39,0,41,0,0,0,0,0 };
S(s39)={ 1, 'e', 40 };
S(s40)={ 0, 0,0,0,0,0,0,0,0,0,0,83,0,83,0,0,0,0,0,0,0,0,0,0,0,0,0 };
S(s41)={ 1, 'o', 42 };
S(s42)={ 1, 'i', 43 };
S(s43)={ 1, 'c', 44 };
S(s44)={ 0, 88,88,88,88,88,88,88,88,88,88,88,88,88,88,88,88,88,88,87,88,
                                                 88,88,88,88,88,88 };
S(s45)={ 1, 'n', 46 };
S(s46)={ 1, 'e', 47 };
S(s47)={ 0, 0,0,0,87,0,0,0,0,48,0,0,0,83,0,0,0,0,0,0,0,0,0,0,0,0,0 };
S(s48)={ 1, 'c', 87 };
S(s49)={ 0, 0,0,0,0,0,0,50,35,0,0,0,83,0,0,0,0,0,51,0,0,0,0,0,0,0,0 };
S(s50)={ 1, 'o', 87 };
S(s51)={ 1, 'a', 52 };
S(s52)={ 1, 'n', 53 };
S(s53)={ 0, 88,88,88,88,81,88,88,88,88,88,88,88,88,88,54,88,88,88,88,88,
                                                 88,88,88,88,88,88 };
S(s54)={ 1, 'i', 82 };
S(s55)={ 0, 0,0,0,0,80,0,0,80,80,0,80,80,0,0,80,0,0,0,0,13,80,80,80,80,
                                                                80,0};

static char     *Suffixes[] =
{
        s0,  s1,  s2,  s3,  s4,  s5,  s6,  s7,  s8,  s9,
        s10, s11, s12, s13, s14, s15, s16, s17, s18, s19,
        s20, s21, s22, s23, s24, s25, s26, s27, s28, s29,
        s30, s31, s32, s33, s34, s35, s36, s37, s38, s39,
        s40, s41, s42, s43, s44, s45, s46, s47, s48, s49,
        s50, s51, s52, s53, s54, s55
};

/*----------------------------------------------------------------------*/

S(p0 )= {1, 0, 0 };
S(p1 )= {0, 0,2,4,6,9,0,0,13,22,0,0,26,29,39,41,45,50,0,53,57,64,
                                                          0,0,0,0,0 };
S(p2 )= {1, 'e', 3 };
S(p3 )= {0, 0,0,81,0,0,0,0,81,0,0,0,0,0,0,0,0,0,0,81,0,0,0,81,0,0,0 };
S(p4 )= {1, 'o', 5 };
S(p5 )= {0, 0,0,0,0,0,0,0,0,0,0,0,87,87,0,0,0,0,0,0,0,0,0,0,0,0,0 };
S(p6 )= {1, 'i', 7 };
S(p7 )= {1, 's', 8 };
S(p8 )= {0, 85,85,85,85,85,85,85, 0,85,85,85,85,85,85,85,85,85,85,85,85,
                                                 85,85,85,85, 0,85 };
S(p9 )= {0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,0,0,0,0,0,0,87,0,0 };
S(p10)= {1, 'u', 11 };
S(p11)= {1, 'i', 12 };
S(p12)= {0, 81,81,81,81,81,81,81,81,81,81,81,81,81,81,81,81,81,81,81,81,
                                                 81, 0,81,81,81,81 };
S(p13)= {0, 14,0,0,0,0,0,0,0,0,0,0,0,0,16,0,0,0,0,0,0,0,0,0,0,19,0 };
S(p14)= {1, 'n', 15 };
S(p15)= {1, 'd', 87 };
S(p16)= {1, 'r', 17 };
S(p17)= {1, 's', 18 };
S(p18)= {1, 'e', 87 };
S(p19)= {1, 'p', 20 };
S(p20)= {1, 'e', 21 };
S(p21)= {1, 'r', 84 };
S(p22)= {0, 0,0,0,0,0,0,0,0,0,0,0,0,86,69,0,0,0,0,0,0,0,0,0,0,0,0 };
S(p23)= {0, 81,0,0,0,0,81,81,0,0,0,0,81,81,0,0,0,0,0,0,0,0,0,0,0,0,0 };
S(p24)= {0, 0,0,0,0,21,0,0,0,0,0,0,0,0,0,0,0,0,25,0,0,0,0,0,0,0,0 };
S(p25)= {1, 'o', 87 };
S(p26)= {1, 'e', 27 };
S(p27)= {1, 'x', 35 };
S(p28)= {1, 'i', 87 };
S(p29)= {0, 30,0,0,0,0,0,0,0,36,0,0,0,0,0,0,0,0,0,0,0,37,0,0,0,0,0 };
S(p30)= {0, 0,0,31,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,33,0,0,0,35,0,0 };
S(p31)= {1, 'r', 32 };
S(p32)= {1, 'o', 83 };
S(p33)= {1, 'h', 34 };
S(p34)= {1, 'e', 82 };
S(p35)= {1, 'i', 82 };
S(p36)= {1, 'n', 35 };
S(p37)= {1, 'l', 38 };
S(p38)= {1, 't', 65 };
S(p39)= {1, 'o', 40 };
S(p40)= {1, 'n', 86 };
```

(Continued on page 38)

More Power Than You Thought Possible

Arity offers the first serious implementation of Prolog for IBM personal computers. Arity/Prolog is a powerful, highly optimized, and extended version of the logic programming language Prolog. Imagine building software applications with a language that solves problems through deduction and logical inference. The task of creating complex programs is much faster and easier, resulting in lower development costs. Arity/Prolog is now in use in a wide range of applications in industry, business, research, and education. The solution—the *Arity/Prolog Interpreter:*

- Source level debugger
- Virtual databases, each with a workspace of 16 megabytes
- Floating-point arithmetic
- String support for efficient text handling



- Interface to assembly language and 'C'
- Text screen manipulation
- Integrated programming shell to MSDOS
- Comprehensive set of evaluable predicates
- Definite clause grammar support

Arity/Prolog Interpreter $495.00

Arity also offers the *Arity/Prolog Compiler and Interpreter,* a sophisticated development environment for building AI applications. Essential for producing fast, serious production code.

Arity/Prolog Compiler and Interpreter $1950.00

The *Arity/Prolog Demo Disk* is available for $19.95. ■ Arity/Prolog products run on the IBM PC, XT, AT, and all IBM compatibles. ■ To order, call (617) 371-2422 or use the order form below.

arity corporation 358 Baker Avenue, Concord, MA 01742

Name ______

Organization ______

Address ______

Quantity	Product	Unit Price	Total Price
	Arity/Prolog Compiler & Interpreter	$1950.00	
	Arity/Prolog Interpreter	$ 495.00	
	Arity/Prolog Demo Disk	$ 19.95	
		Subtotal	
		MA residents add 5% sales tax	
		Total Amount	

☐ Enclosed is a check or money order to Arity Corporation

☐ Please bill my ☐ Mastercard ☐ Visa ☐ American Express

Account # ______

Valid ____/____ to ____/____ ______ signature

☐ Please send me more information about Arity and Arity/Prolog

arity 358 Baker Avenue, Concord, MA 01742

M-AD-05

Circle **no. 16** on reader service card.

TURBO *Power*™

UTILITIES

FOR

Turbo Pascal™ PROGRAMMERS

Improve Code Performance
Find Subtle Bugs
Automate Tedious Tasks

Supports Turbo Pascal 2.0 & 3.0
IBM PC/XT/AT & True Compatibles
PCDOS 2.X & 3.X
192K RAM DSDD Drive

If You Really Use Your Pascal Compiler You Need These Tools!

- Pretty Printer
- Pascal Structure Analyzer
- Execution Profiler
- Execution Timer

Advanced Text Processing & Command Automation

- Pattern Replacer
- Difference Finder
- Command Builder
- File Finder
- Super Directory

Where Else Can You Get 500K of Integrated, Useful, Tested, Fully Documented Source Code for $95?

- 140 Page Printed User Manual
- Quick Reference Card
- Detailed Programmer's Manual on Disk
- Advanced Subroutine Library
- Complete Turbo Pascal Source Code
- Compiled Programs Also Included
- 6 Bonus Utilities!
- Tax & U.S. Postage Included
- Executable only version $55

MC/Visa Orders TOLL FREE
(USA) 800-538-8157 x830
(CAL) 800-672-3470 x830
Brochures, Questions, PO's call 408-378-3672
Checks or Money Orders
TurboPower Software
478 W. Hamilton Ave., Suite 196
Campbell, CA 95008 U.S.A.

INTERNATIONAL REPRESENTATIVES —
Switzerland: Software Haus 064 51 26 51
Japan: Southern Pacific Ltd 045 314 9514
England: The Core Store 0606 45420

Turbo Pascal Is a Trademark of Borland International

Circle **no. 81** on reader service card.

C Chest (Listing continued, text begins on page 20)

Listing Two

```
S(p41)= {0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,42,43,0,0,0,0 };
S(p42)= {1, 't', 87 };
S(p43)= {1, 'e', 44 };
S(p44)= {1, 'r', 86 };
S(p45)= {1, 's', 46 };
S(p46)= {1, 'e', 47 };
S(p47)= {1, 'u', 48 };
S(p48)= {1, 'd', 49 };
S(p49)= {1, 'o', 83 };
S(p50)= {1, 'u', 51 };
S(p51)= {1, 'a', 52 };
S(p52)= {1, 'd', 87 };
S(p53)= {0, 0,0,0,0,54,0,0,0,0,0,0,0,0,55,0,0,0,0,0,56,0,0,0,0,0 };
S(p54)= {1, 'm', 28 };
S(p55)= {1, 'm', 18 };
S(p56)= {0, 0,87,0,0,0,0,0,0,0,0,0,0,0,0,20,0,0,0,0,0,0,0,0,0,0 };
S(p57)= {0, 0,0,0,0,0,0,0,58,0,0,0,0,0,0,0,0,0,61,0,0,0,0,0,0,0,0 };
S(p58)= {1, 'e', 59 };
S(p59)= {1, 'r', 60 };
S(p60)= {1, 'e', 87 };
S(p61)= {0, 62,0,0,0,0,0,0,0,68,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 };
S(p62)= {1, 'n', 63 };
S(p63)= {1, 's', 23 };
S(p64)= {1, 'n', 66 };
S(p65)= {1, 'i', 83 };
S(p66)= {0, 85,85,85,70,85,85,85,85, 0,85,85,85,85,85,85,85,85,85,
                                        85,85,85,85,85,85,85,85 };
S(p67)= {1, 'r', 87 };
S(p68)= {0, 81,0,0,0,0,81,0,0,0,0,0,0,0,0,0,0,0,0,0,81,0,0,0,0,0 };
S(p69)= {0, 85,85,85,85,85,85,85,85,85,85,85,85,85,85,85,85,85,85,
                                        85,24,85,85,85,85,85,85 };
S(p70)= {1, 'e', 67 };


static char     *Prefixes[] =
{
        p0,  p1,  p2,  p3,  p4,  p5,  p6,  p7,  p8,  p9,
        p10, p11, p12, p13, p14, p15, p16, p17, p18, p19,
        p20, p21, p22, p23, p24, p25, p26, p27, p28, p29,
        p30, p31, p32, p33, p34, p35, p36, p37, p38, p39,
        p40, p41, p42, p43, p44, p45, p46, p47, p48, p49,
        p50, p51, p52, p53, p54, p55, p56, p57, p58, p59,
        p60, p61, p62, p63, p64, p65, p66, p67, p68, p69,
        p70
};

/*======================================================================
 *
 *              MSC DEBUGGING ROUTINES (INCLUDING A MAIN MODULE):
 */

#ifdef DEBUG
bprint( b , e )
ATYPE   *b, *e;
{
        /*      Print all characters between b and e inclusive    */

        while( b <= e && *b )
                putchar( *b++ & 0x7f );
}

/*-------------------------------------------------------------------*/

char    *sgets( prompt, buf, maxch )
char    *prompt, *buf;
{
        /* Print prompt to stdout, then fill buf fm stdin. Get at most
         * maxch characters. Return pointer to string terminator (ie. the
         * null at the end of the string) or 0 on end of file.
         */

        register int    c;
        register char   *start;

        start = buf;
        printf( prompt );

        while( (c = getchar()) != EOF && c != '\n' && --maxch > 0 )
                *buf++ = c;

        *buf = 0;

        return (c == EOF && start == buf) ? (char *)0 : buf ;
}

/*-------------------------------------------------------------------*/

phyphen( p )
ATYPE *p;
{
        /*      Print a word with all the hyphens visible
         */

        printf("<");

        for( ; *p ; putchar(*p++ & 0x7f) )
                if( HAS_HYPHEN(*p) )
                        putchar('-');

        printf(">\n");
}

/*------------------------------------------------------------------*/

main(argc, argv)
int     argc;
char    **argv;
{
        char    buf[132], *p;

        if( argc > 1 )
                Debug = 1;

        while( p = sgets( "word: ", buf, 132 ) )
        {
                if( !*buf )
                        printf("\n");

                else if( p-- > buf )
                {
                        hyphen( buf, p );
                        putchar('\t');
                        phyphen( buf );
                }
        }
}

#endif
```

End Listing Two

Listing Three

```
+------------------------------------------------------------------------+
|    Diagnostic output from hyphen.c when compiled with the DEBUG #defined:  |
+------------------------------------------------------------------------+

word: be/cxxxe/able      <be-cxxxe-able>
word: be/hxxxh/able      <be-hxxxh-able>
word: be/sxxxi/able      <be-sxxxi-able>
word: be/wxxxk/able      <be-wxxxk-able>
word: com/xxxl/able      <com-xxxl-able>
word: con/xxxo/able      <con-xxxo-able>
word: dis/xxxu/able      <dis-xxxu-able>
word: equi/xxv/able      <equi-xxv-able>
word: ex/xxxxw/able      <ex-xxxxw-able>
word: hand/xxxxxx/able   <hand-xxxxxx-able>
word: horse/xxxxy/able   <horse-xxxxy-able>
word: hyper/xxxnt/able   <hy-per-xxxnt-able>
word: im/xxxxxxrt/able   <im-xxxxxxrt-able>
word: in/xxxxxxion/ary   <in-xxxxxxion-ary>
word: inter/xxxxen/ary   <in-ter-xxxxen-ary>
word: intro/xxxxx/nary   <in-tro-xxxxx-nary>
word: lexi/xxxxxxx/cal   <lex-i-xxxxxxx-cal>
word: macro/xxxxa/cate   <mac-ro-xxxxa-cate>
word: mathe/xxxxe/cate   <math-e-xxxxe-cate>
word: maxi/xxxxxi/cate   <max-i-xxxxxi-cate>
word: mini/xxxxxo/cate   <min-i-xxxxxo-cate>
word: multi/xxxxu/cate   <mul-ti-xxxxu-cate>
word: non/xxxxxxy/cate   <non-xxxxxxy-cate>
word: out/xxxxxxx/cial   <out-xxxxxxx-cial>
word: over/xxxxx/cious   <over-xxxxx-cious>
word: pseudo/xx/scious   <pseu-do-xx-scious>
word: quad/xxxxx/cient   <quad-xxxxx-cient>
word: semi/xxxxxx/dent   <semi-xxxxxx-dent>
word: some/xxxxxxx/ful   <some-xxxxxxx-ful>
word: sub/xxxxxxxl/ize   <sub-xxxxxxxl-ize>
word: super/xxxxa/late   <su-per-xxxxa-late>
word: there/xxxxe/late   <there-xxxxe-late>
word: trans/axxxi/late   <trans-ax-xxi-late>
word: trans/fxxxo/late   <trans-fxxxo-late>
word: trans/gxxxu/late   <trans-gxxxu-late>
word: trans/lxxxy/late   <trans-lxxxy-late>
word: trans/mxxxx/less   <trans-mxxxx-less>
word: tri/axxxxxxxx/ly   <tri-ax-xxxxxxx-ly>
word: tri/fxxxxxxxx/ly   <tri-fxxxxxxxx-ly>
word: tri/uxxxxxxxx/ly   <tri-ux-xxxxxxx-ly>
word: under/xxxxx/ment   <un-der-xxxxx-ment>
word: un/xxxxxxxx/ness   <un-xxxxxxxx-ness>
word: /xxxxx/ogy         <xxxxx-ogy>
word: /xx/rapher         <xx-rapher>
word: /xxx/raphy         <xxx-raphy>
word: /xxx/scope         <xxx-scope>
word: /xx/scopic         <xx-scopic>
word: /xxx/scion         <xxx-scion>
word: /xx/sphere         <xx-sphere>
word: /xxxxx/tal         <xxxxx-tal>
word: /xxxx/tial         <xxxx-tial>
word: /xxxx/tion         <xxxx-tion>
word: /xx/tional         <xx-tion-al>
word: /xxxx/tive         <xxxx-tive>
word: /xxxx/ture         <xxxx-ture>
word: /xxxxxl/ing        <xxxxxl-ing>
word: /xxxxl/ing         <xxxxl-ing>
word: /xxxg/gling        <xxxg-gling>
word: /xxxa/bling        <xxxa-bling>
```

(Continued on next page)

Is $19.97 too much to spend to avoid a million dollar mistake?

Business Software magazine is a valuable resource that helps you squeeze every penny's worth of potential out of your computer system. It is written for the business person and decision maker who wants practical, "hands-on" software solutions to real problems, written in a straightforward, jargon-free language. Every month, **Business Software** brings you: case studies of business computer applications; company profiles, including who is using what software and why; software tutorials that explain what the manual left out; information on the latest products, as well as the longtime favorites, all from a business perspective.

To subscribe, please fill out this coupon and return it to:
Business Software,
P.O. Box 27975,
San Diego, CA 92128

Name ____________________
Address ____________________
City ____________________
State __________ Zip __________
Bill my ☐ VISA ☐ M/C
☐ Bill me later ☐ $19.97
Card No. ____________________
Expiration date ____________________
Signature ____________________
Please allow 6–9 weeks for address change to take effect 3031

3043

GROWING OLD?

...waiting for **C** programs to compile and link?

Use **C-terp**
the **complete** C interpreter

This is the product you've been waiting (and waiting) for!

Increase your productivity and avoid agonizing waits. Get instant feedback of your C programs for debugging and rapid prototyping. Then use your compiler for what it does best...compiling efficient code ...slowly.

C-terp Features

- Full K&R C (no compromises)
- Complete built-in screen editor-- no half-way house, this editor has everything you need such as multi-files, inter-file move and copy, etc. etc. For the ultimate in customization, editor source is available for a slight additional charge of $98.00.
- Fast--Linking and semi-compilation are breath-takingly fast. (From edit to run completion in a fraction of a second for small programs.)
- Convenient--Compiling and running are only a key-stroke or two away. Errors direct you back to the editor with the cursor set to the trouble spot.
- Object Module Support– Access functions and externals in object modules produced by your compiler. **New:** We are now supporting **Microsoft** 3.0, **Mark Williams** & **Aztec C** in addition to C.I. C86 & Lattice.
- Complete Multiple Module Support.
- Symbolic Debugging-- Set breakpoints, single-step, and directly execute C expressions.
- Many more features including batch mode and 8087 support.
- **Price: $300.00 (Demo $45.00) MC, VISA**

Price of demo includes documentation & shipping within U.S. PA residents add 6% sales tax. Specify compiler.

- C-terp runs on the IBM PC (or compatible) under DOS 2.x with a suggested minimum of 256Kb of memory. It can use all the memory available.

GIMPEL SOFTWARE
3207 Hogarth Lane • Collegeville, PA 19426
(215) 584-4261

*Trademarks: C86 (Computer Innovations), Lattice (Lattice Inc.) IBM (IBM Corp.), C-terp (Gimpel Software), Microsoft (Microsoft), Aztec (Manx)

C Chest

(Listing continued, text begins on page 20)

Listing Three

```
word: /xxxck/ling          <xxxck-ling>
word: /xxxg/kling          <xxxg-kling>
word: /duck/ing            <duck-ing>
word: /fit/ting            <fit-ting>
word: /hiss/ing            <hiss-ing>
word: /swell/ing           <swell-ing>
word: /fic/tion            <fic-tion>
word: /fic/tional          <fic-tion-al>
word: /fic/tionalize       <fic-tion-al-ize>
word: /fic/tionalized      <fic-tion-al-ized>
word: /fic/tionalizes      <fic-tion-al-izes>
word: /uuuccuuu/           <uuuc-cuuu>
word: /uuuquuuu/           <uuu-quuuu>
word: /uuuckuuu/           <uuuck-uuu>
word: /uuuccuuu/           <uuuc-cuuu>
word: /uuuchchuu/          <uuuch-chuu>
word: /uuughghuu/          <uuugh-ghuu>
word: /uuuphphuu/          <uuuph-phuu>
word: /uuushshuu/          <uuush-shuu>
word: /uuuththuu/          <uuuth-thuu>
word: /uuuxxxxxx/          <uuux-xxxxx>
word: /uuullaxxxx/         <uuul-lax-xxx>
word: /uuussexxxx/         <uuus-sex-xxx>
word: /uuufteee/e          <uuuf-teeee>
word: /uuuxxer/            <uuux-xer>
word: /uuuxxer/s           <uuux-xers>
word: /uuuxxag/e           <uuux-xage>
word: /uuuxxage/s          <uuux-xages>
word: /uuuxxest/           <uuux-xest>
word:
word:    <Test repetetive suffix and prefix removal:>
word:
word: disiminnonoverun/en/aryfulizelesslymentableness
                <dis-im-in-non-over-un-en-ary-ful-ize-less-ly-ment-able-ness>
word:
word: super/califragilisticexpialido/cious
                <su-per-califragilis-ticex-pialido-cious>
word:
word:    <All words following this point test exceptions to>
word:    <the hyphenation rules and shouldn't be hyphenated.>
word:
word: /uuussx/             <uuussx>
word: /uuullx/             <uuullx>
word: /uuuller/            <uuuller>
word: /uuuller/s           <uuullers>
word: /uuubluu/            <uuubluu>
word: /uuubruu/            <uuubruu>
word: /uuucluu/            <uuucluu>
word: /uuucruu/            <uuucruu>
word: /uuuchluu/           <uuuchluu>
word: /uuuchruu/           <uuuchruu>
word: /uuudguu/            <uuudguu>
word: /uuudruu/            <uuudruu>
word: /uuufluu/            <uuufluu>
word: /uuufruu/            <uuufruu>
word: /uuughtuu/           <uuughtuu>
word: /uuugluu/            <uuugluu>
word: /uuugruu/            <uuugruu>
word: /uuuknuu/            <uuuknuu>
word: /uuulkuu/            <uuulkuu>
word: /uuulquu/            <uuulquu>
word: /uuunchuu/           <uuunchuu>
word: /uuunkuu/            <uuunkuu>
word: /uuunxuu/            <uuunxuu>
word: /uuuphruu/           <uuuphruu>
word: /uuupluu/            <uuupluu>
word: /uuupruu/            <uuupruu>
word: /uuurkuu/            <uuurkuu>
word: /uuuspuu/            <uuuspuu>
word: /uuusquu/            <uuusquu>
word: /uuutchuu/           <uuutchuu>
word: /uuutruu/            <uuutruu>
word: /uuuthruu/           <uuuthruu>
word: /uuuwhuu/            <uuuwhuu>
word: /uuuwluu/            <uuuwluu>
word: /uuuwnuu/            <uuuwnuu>
word: /uuuwruu/            <uuuwruu>
word: /uuufter/            <uuufter>
word: /uuufter/s           <uuufters>
word: /uuuftag/e           <uuuftage>
word: /uuuftage/s          <uuuftages>
word: /uuuftest/           <uuuftest>
word: /uuulder/            <uuulder>
word: /uuumper/            <uuumper>
word: /uuunder/            <uuunder>
word: /uuunger/            <uuunger>
word: /uuunser/            <uuunser>
word: /uuunter/            <uuunter>
word: /uuurger/            <uuurger>
word: /uuurner/            <uuurner>
word: /uuurter/            <uuurter>
word: /uuuster/            <uuuster>
```

End Listings

Another in a series of productivity notes on MS-DOS™ software from UniPress.

TEXT EDITING

UNIPRESS EMACS™

Subject: Multi-window full screen editor.

Multiple windows allow several files (or portions of the same file) to be edited simultaneously. Programmable through macros and the built-in compiled MLISP™ extension language.

Features:

- *Famed Gosling version.*
- *Extensible through macros and the built-in compiled MLISP extension language.*
- *Dozens of source code MLISP functions; including C, Pascal, LISP and MLISP syntax checking.*
- *EDT and simple WordStar™ emulation modes.*
- *MS-DOS commands can be executed with output placed in an EMACS window.*
- *Run a compile on your program and EMACS will point to any errors for ease of debugging.*
- *EMACS runs on the IBM-PC™ (XT/AT), TI-PC,™ DEC Rainbow 100+,™ HP-150™ or any other MS-DOS machine. Requires at least 384K.*

Price:

EMACS binary	*$325*
EMACS source	*995*
One month trial	*75*

Also available for UNIX™ and VMS.™
Call for pricing.

ISAM FILE SYSTEM REPORT WRITER & QUERY LANGUAGE

PHACT™

Subject: Powerful keyed file access, report generator and query language for MS-DOS.

PHACT Isam is a keyed B+ tree file manager providing easy access to and manipulation of records in a database. PHACT-report produces formatted reports from PHACT databases. PHACT-query is a high level tool for manipulating data in PHACT databases.

Features:

- *Supports fixed and variable length records (1-9999 bytes).*
- *Up to 9 alternate indices are supported.*
- *Record locking allows multiple simultaneous updates.*
- *Records can be accessed on full or partial key.*
- *Includes full Lattice™ linkable library and high-level functions.*
- *PHACT-report, an easy-to-use command language for formatting reports from existing PHACT databases.*
- *PHACT-query can be used for making ad hoc and "canned" inquiries into a PHACT database.*

Price:

PHACT ISAM	*single-user*	*$295*
	multi-user	*450*
PHACT-report	*single-user*	*165*
	multi-user	*275*
PHACT-query	*single-user*	*195*
	multi-user	*495*

Source code available. Call for terms.
Also available for UNIX and VMS.
Call for pricing.

COMMUNICATIONS

PCworks™

New from UniPress!

Subject: Communication network for PCs.

PCworks connects PCs to a variety of computers and information services.

Features:

- *PCworks connects your PC to other PCs/Macintoshes™ or UNIX/VMS systems for file transfer and command serving.*
- *Menu driven for ease of use. Telephone numbers and other parameters can be stored.*
- *Transfer ASCII or binary files, send/receive electronic mail, or submit and print jobs in conjunction with UNIX or VAX™/VMS hosts.*
- *PC files can be printed on another system.*
- *Terminal Emulators include: ANSI, TTY, VT100/52.*
- *Protocol: error-free data transfers utilizing packet sequence numbers and checksums, or XMODEM protocol.*
- *Supports many modems/autodialers at a wide range of speeds.*

Price:

PCworks	*$195*
The Connectable Networks™	*Call for pricing.*

*For our **Free Catalogue** and more information on these and other software products, call or write: UniPress Software, Inc.; 2025 Lincoln Hwy., Edison, NJ 08817. Telephone: (201) 985-8000. **Order Desk: (800) 222-0550 (Outside NJ).** Telex 709418. European Distributor: Modulator SA, Switzerland Telephone: 41 31 59 22 22, Telex: 911859*

OEM terms available. Mastercard/Visa accepted.

See our ad in the next issue for more MS-DOS products, including Lattice C compiler and Ps-Make.

Trademarks of: UniPress EMACS and MLISP, UniPress Software, Inc.; WordStar, MicroPro Intl. Corp.; MS-DOS and IBM-PC, IBM.; TI-PC, Texas Instruments; PHACT, PHACT Associates; Lattice, Lattice, Inc.; UNIX, AT&T Bell Laboratories; VAX/VMS, DEC Rainbow 100+, Digital Equipment Corp; HP-150, Hewlett-Packard Co. MACLine, UniHost, PCworks and The Connectables Network, Touchstone Software, Corp; Macintosh, Apple Computer, Inc.

UniPress Software

Circle **no. 77** on reader service card.

A Threaded-Code Microprocessor Bursts Forth

Subroutines Without Performance Anxiety

by Leo Brodie

Subroutine calls in one clock cycle are one advantage as Forth goes to silicon.

The Novix NC4000 processor has arrived, and it threatens to shake most of our beliefs about microprocessor design. What implications does this new technology have for the art of programming? This article will explain the techniques used by Forth and by the NC4000 microprocessor, and explore the impact of this technology on software design.

Subroutines: The More the Better

Most programmers recognize the importance of dividing a large, complex program into a group of smaller, manageable pieces. The use of smaller "modules" is advantageous for the following reasons:

- smaller pieces of code are easier to think about, and therefore easier to design and to write;
- smaller pieces of code are easier to debug; the likelihood of introducing a bug is reduced, and, if one does appear, it will be easier to find;
- different parts of a program may be assigned to different programmers;
- subroutines can be used to hide information about things that might change in subsequent revisions; if a change must be made to a program, only the subroutine has to be changed.

Unfortunately, most conventional high-level languages (HLLs) are not optimized for the invocation of subroutines. One problem is the loss of time caused by saving registers before a jump to a subroutine and restoring them afterwards. This has been estimated as 40% of program execution time. Even more time-consuming is the invisible but significant code needed to pass arguments to and from the subroutine. At the machine level as well, processors are not generally optimized for invoking subroutine calls.

For these reasons, although the use of many, small subroutines is an ideal, nevertheless, in practice, programmers tend to writer fewer, larger subroutines. When efficiency is at stake, it's hard to appreciate the elegance of subroutines.

There is a high-level programming language optimized for subroutines: Forth. And now, a single-chip microprocessor is available that applies Forth's optimization techniques directly to a silicon architecture. The result is a machine that can execute high-level subroutine calls and returns in a single clock cycle!

Threaded Code

Let's begin by examining what the inherent architecture of Forth is, regardless of which processor it runs on. Forth is implemented using a technique called "threaded code." Suppose we define a word in Forth called PROCESS, consisting of three operations.

```
: PROCESS        STEP1  STEP2  STEP3  ;
```

In this definition, the name of the word (subroutine) is PROCESS. It consists of three lower-level subroutines. The colon begins the definition, while the semicolon ends it.

Figure 1 (page 44) shows what this definition looks like when compiled into memory on an emulated Forth system. Such a definition consists of the following parts:

- The name field contains an ASCII representation of the word's name, so that the word can subsequently be found and executed.
- The link field points to the previous definition in the linked list of dictionary entries. (A search through the dictionary for a given word begins with the most recently defined word, and, using the link field for each word, proceeds backwards through the chain or chains of words.)
- The code pointer field points to machine-executable code that is generic to each "family" of definitions. As yet we have only discussed colon definitions; we will introduce other types later.
- The remainder of the definition is a list of addresses of other routines. Depending on the processor, these addresses are usually 16- or 32-bit numbers. In this example, the addresses of the routines STEP1, STEP2, and STEP3 are listed. At the end of the list is the address of another routine called EXIT, which is compiled by the semicolon and which terminates the definition when it is executed.

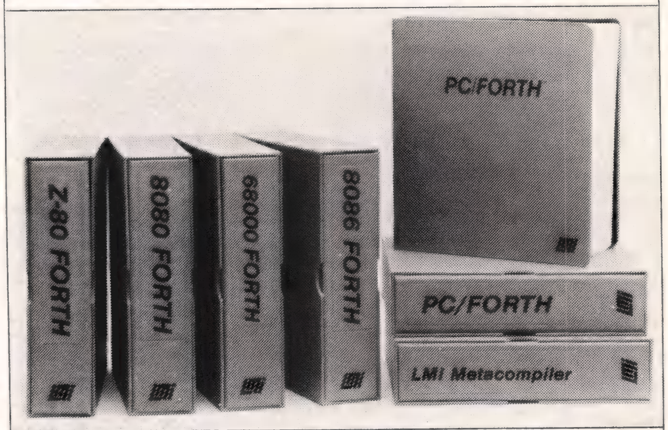
The unusual thing about threaded code is the absence of any CALL instructions—*everything* is a subroutine. Definitions consist simply of addresses of other definitions. As we'll see, even data structures are defined as dictionary entries, so that their addresses appear in definitions when referenced.

If you expand your imagination a bit, you can view these addresses as actual *instructions*, since each address is unique. This is exactly the way they are viewed by the Forth machine, and in particular, by the routine called the "address interpreter," also known as NEXT.

It is the function of the address interpreter to execute each "Forth instruction" in turn. This function is analogous to the operation of a machine processor wherein each opcode is interpreted and the program counter advanced. An "interpreter pointer," called **I** (usually a register in the real processor) operates analogously to the processor's program counter: during execution of any Forth instruction, I points to the next instruction to be executed; it is incremented by the address interpreter.

To begin with the simplest case possible, let's assume that each of the definitions STEP1, STEP2, and STEP3 is defined in machine code—they are not high level Forth definitions. (Forth allows routines to be coded in high-level code or in machine code without distinction; in practice, however, only a handful of time-critical or hardware-dependent routines must be machine-coded.) Here's what happens, in somewhat simplified form for now. I is pointing to the first address in the definition of PROCESS; this is the address of STEP1. The address interpreter, NEXT, first advances I to the next cell (the one containing the address of STEP2), then jumps to STEP1. STEP1 is a machine code definition, executed in the normal way by the

TOTAL CONTROL
with *LMI FORTH*™



For Programming Professionals:
an expanding family of compatible, high-performance, Forth-83 Standard compilers for microcomputers

For Development:
Interactive Forth-83 Interpreter/Compilers
- 16-bit and 32-bit implementations
- Full screen editor and assembler
- Uses standard operating system files
- 400 page manual written in plain English
- Options include software floating point, arithmetic coprocessor support, symbolic debugger, native code compilers, and graphics support

For Applications: Forth-83 Metacompiler
- Unique table-driven multi-pass Forth compiler
- Compiles compact ROMable or disk-based applications
- Excellent error handling
- Produces headerless code, compiles from intermediate states, and performs conditional compilation
- Cross-compiles to 8080, Z-80, 8086, 68000, and 6502
- No license fee or royalty for compiled applications

Support Services for registered users:
- Technical Assistance Hotline
- Periodic newsletters and low-cost updates
- Bulletin Board System

Call or write for detailed product information and prices. Consulting and Educational Services available by special arrangement.

LMI ***Laboratory Microsystems Incorporated***
Post Office Box 10430, Marina del Rey, CA 90295
Phone credit card orders to: ***(213) 306-7412***

Overseas Distributors.
Germany: Forth-Systeme Angelika Flesch, D-7820 Titisee-Neustadt
UK: System Science Ltd., London EC1A 9JX
France: Micro-Sigma S.A.R.L., 75008 Paris
Japan: Southern Pacific Ltd., Yokohama 220
Australia: Wave-onic Associates, 6107 Wilson, W.A.

Circle no. 55 on reader service card.

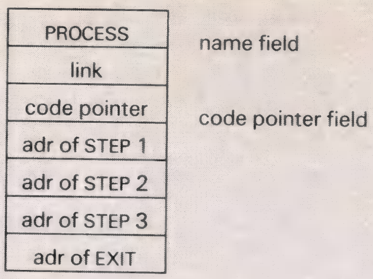


Figure 1
A compiled definition in the dictionary of an emulated Forth system

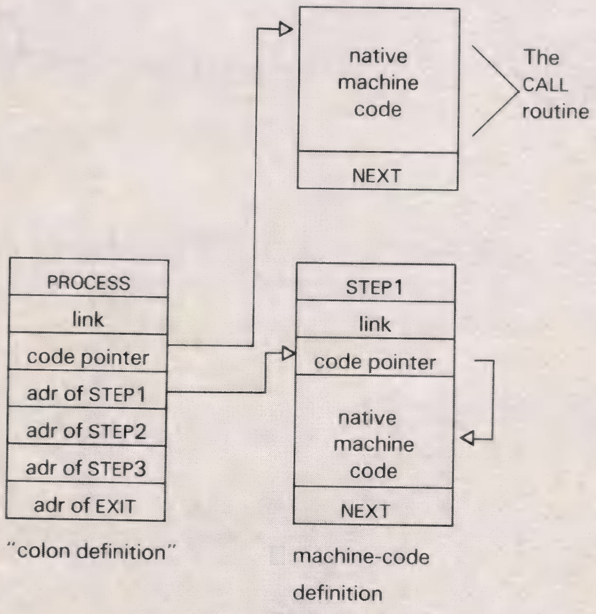


Figure 2
The code pointer of an assembler definition (e.g., STEP1) points to its own definition. The code pointer of a high-level definition (e.g., PROCESS) points to a prologue routine called CALL.

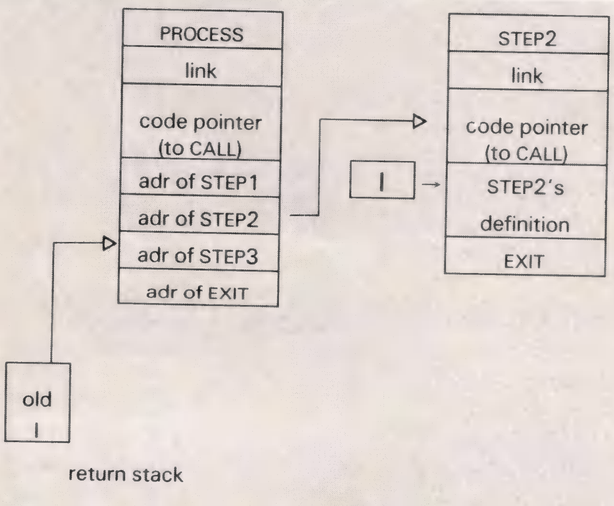


Figure 3
When STEP2's CALL routine is invoked, it saves I (the interpreter pointer) on the return stack, puts the address of the body of STEP2's definition in I, then invokes NEXT.

underlying processor. The final piece of code in STEP1 is NEXT once again (either in-line or as a jump instruction, depending on whether performance is a more or less important consideration than memory usage in the particular hardware configuration). This second NEXT increments I to the third cell, then jumps to STEP2, and so on.

Each machine-coded definition (called a primitive) must end by executing the code for NEXT to keep the Forth machine running.

In the preceding discussion, we omitted one important detail. The address interpreter does not jump to the address of each new instruction, but rather to the *code pointed to by the code field* of each new instruction—one added level of indirection. Figure 2 (at left) shows this relationship.

In the case of a machine code definition, the code field points to the body of the definition itself, where the actual machine instructions have been compiled. In the case of all colon (high-level) definitions, however, the code field points to a single routine in memory that we will name CALL. Essentially this routine handles the job of nesting definitions, so that one high-level word may invoke another word, which in turn invokes another, and so on.

Here is what CALL does, step by step. Using our previous example, let's now suppose that STEP2 has been defined as a colon, or high-level, definition. We are executing PROCESS, and have just completed STEP1. I (the interpreter pointer) is pointing to the address of STEP2. The address interpreter advances the pointer in anticipation of the return, then gets the address of STEP2. Using this address, it jumps to the address pointed to by the code pointer field of STEP2. Since STEP2 is a high-level definition, the routine at this address is CALL. This routine saves I on a special stack for return addresses called the return stack. It then places into I the address of the beginning of the instruction list in STEP2 (see Figure 3, at left). This done, CALL executes NEXT. Thus the Forth machine continues as before, but has nested in a level.

The opposite of the CALL function is EXIT, which terminates each high-level definition. When the EXIT in STEP2 is encountered, it pops the top of the return stack back into the interpreter pointer, thus resuming execution of PROCESS at the address of STEP3.

Because of the extra level of indirection offered by the code pointer field, the architecture we've described has been called indirect threaded code (ITC).

The value of the code pointer field is that each word knows what kind of procedure it is, and how to invoke itself. This is what makes it possible in Forth to design a routine first in high-level code, then to rewrite it in machine code without changing any code elsewhere that invokes the routine.

Each family of data structures in Forth uses a unique prologue routine, pointed to by the code fields of all members of that family. A simple example is the constant, as illustrated by Figure 4 (page 45). A Forth constant is a dictionary entry, possessing the same type of header structure as all dictionary entries. Its body consists of a code pointer for all constants, along with the value of the constant. The code pointer points to a routine, which we will call "do-constant," whose job it is to fetch the value from the body of the constant, then invoke NEXT.

The Forth Engine in Silicon

In indirect threaded code implementations of Forth as just described, the invoking of words is elegant and more efficient than the calling of subroutines in other high-level languages. However, because conventional microprocessors are not optimized for ITC, running the Forth machine involves some performance overhead. This overhead consists of the time spent executing NEXT after each machine instruction, plus the time spent for each call and return. Running high-level Forth versus straight machine code can increase execution time by as much as 100 percent (which is still better than other high-level languages).

A primary goal of the NC4000 architecture is to allow the execution of threaded code with as little overhead as possible. This goal is achieved first by eliminating NEXT as a software routine. Instead, NEXT has become the *background* to the chip's operation. The overhead of NEXT is reduced from many clock cycles to less than one (see Figure 5, at right).

Secondly, the NC4000 reduces the normal overhead of the CALL operation. In fact, it achieves this goal to the utmost degree possible: a subroutine call executes in a *single clock cycle*. Compare this with the 15 clocks required by the 8086 to effect a machine-level subroutine call. A high-level language such as C will require still more instructions to effect a jump to subroutine.

How does the NC4000 achieve these results? One of the bits (the MSB) of the 16-bit opcode is reserved exclusively to indicate a subroutine call. For all ordinary machine instructions, this bit is one; but if this bit is zero, the NC4000 effects a call to the subroutine whose address is specified by the remaining 15 bits. Thus, colon definitions appear as compiled object code as shown in Figure 6 (at right).

As you can see, compiled object code for the NC4000 consists of a mixture of actual machine instructions and single-instruction subroutine calls to definitions. In other words, the NC4000 executes high-level Forth threaded code as its native machine language.

Of actual machine instructions the NC4000P boasts over 170 combinations, not including permutations of register addressing. These include the usual Forth primitives for data-stack manipulation, arithmetic, memory, register, and I/O fetches and stores, jumps and loops—even multiply, divide, and square root steps.

The use of the high-order bit enables the chip's logic to decode each instruction early in the clock cycle. If the instruction is a CALL, then the address represented by the remaining 15 bits will appear on the address bus in time for the appropriate instruction (in the called routine) to be latched and executed on the next cycle.

The NC4000 does even better with the EXIT routine. Another of the 16 bits in the opcode (octal 40) is reserved exclusively to indicate the return operation. The return can occur simultaneously with other operations of the

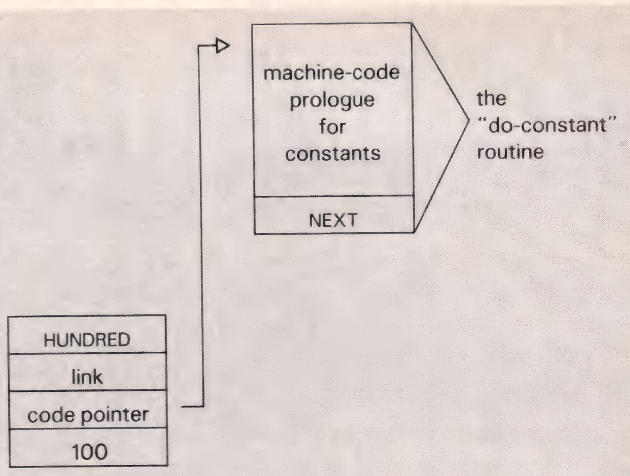

machine-code prologue for constants
NEXT
the "do-constant" routine
HUNDRED
link
code pointer
100

Figure 4
A Forth constant is a normal dictionary entry whose code pointer points to a routine common to all constants.

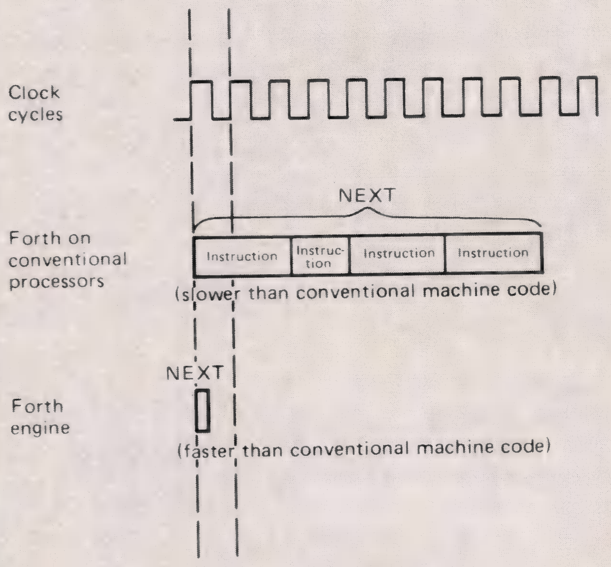

Clock cycles
NEXT
Forth on conventional processors
Instruction
Instruction
Instruction
Instruction
(slower than conventional machine code)
NEXT
Forth engine
(faster than conventional machine code)

Figure 5
Comparison of overhead for NEXT

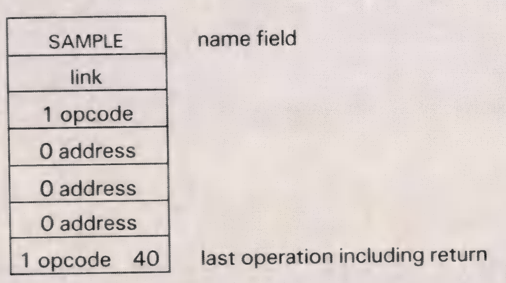


Figure 6
A definition list on the NC4000.

Write it once!

MasterFORTH

Portable programming environment

Whether you program on the **Macintosh**, the **IBM PC**, an **Apple II** series, a **CP/M** system, or the **Commodore 64**, your program will run unchanged on all the rest. If you write for yourself, MasterFORTH will protect your investment. If you write for others, it will expand your marketplace.

IBM™

MasterFORTH is a state-of-the-art implementation of the Forth computer language. Forth is interactive – you have immediate feedback as you program, every step of the way. Forth is fast, too, and you can use its built-in macro assembler to make it even faster. MasterFORTH's relocatable utilities, transient definitions, and headerless code let you pack a lot more program into your memory. The resident debugger lets you decompile, breakpoint, and trace your way through most programming problems. A string package, file interface, and full screen editor are all standard features.

CP/M™

MasterFORTH exactly matches the Forth-83 Standard dialect described in *Mastering Forth* by Anderson and Tracy (Brady, 1984). The standard package includes the book and over 100 pages of supplementary documentation.

MasterFORTH standard package	
Macintosh	$125
IBM PC and PC Jr. (MS DOS 2.1)	125
Apple II, II+, IIe, IIc (DOS 3.3)	125
CP/M 2. x (IBM 3740 8")	125
Commodore 64	100
Extensions	
Floating Point (1984 FVG standard)	$60
Graphics (Apple II series)	60
Module relocator (with utility sources)	60
Printed source listing (each)	35
Publications	
Mastering Forth (additional copies)	$18
Thinking Forth by Leo Brodie	16
Forth-83 International Standard	15
Rochester Bibliography, 2nd ed.	15
1984 Rochester Conference	25
1984 FORML Conference	25

MICROMOTION

12077 Wilshire Blvd., #506
Los Angeles, CA 90025
(213) 821-4340

Circle **no. 70** on reader service card.

ALU. In a single clock cycle, a word can perform its last operation and return to the word that invoked it. Thus, in most cases, there is *no overhead* for the EXIT operation.

One consequence is that program space is limited to 32K addresses. Since the NC4000 uses *word addresses*, not byte addresses, the program may occupy 64K bytes of memory. Given the extreme compactness of threaded-code systems in general, and of this instruction set in particular (each instruction can perform up to five operations simultaneously), 64K of program memory will contain a considerable amount of code compared to traditional systems. Since memory operators use full 16-bit addresses, program size may be further compacted by placing data buffers and tables above the 32K word boundary instead of in the program region. For those who still aren't satisfied, Novix is currently considering development of a 32-bit version of the NC4000 architecture.

A necessary ingredient in this design is the implementation of a return stack in hardware. The return stack can be expressly manipulated by the programmer, but it is also controlled automatically by the CALL and EXIT operations of the processor. The NC4000 features a unique parallel architecture that allows the return stack to operate simultaneously with other CPU operations.

Notice in Figure 6 that the code field is no longer needed, because Forth is the native machine code of the NC4000. The CALL routine, for instance, which serves as the prologue for colon definitions, now exists in processor logic, and is invoked whenever bit 15 is zero. In the case of constants, the pointer to the old "do-constant" routine can now be replaced by a literal-fetch instruction.

In fact, each dictionary entry, whether a definition, a constant, variable, array, or what-have-you, is most efficiently compiled as a definition and directly executed. Thus, the Novix processor uses "direct-threaded-code" (DTC), not ITC.

To summarize, the NC4000 realizes the Forth machine directly in silicon logic, without even resorting to microcode. As a result, it supports a subroutine call and return with an overhead of one clock cycle in most instances, and never more than two. Comparable CALL and RET instructions on the 8086 require a total of 34 clock cycles (and each 8086 cycle is two to five times slower than that of the Novix prototype).

The Software Implications

What does all of this mean to the programmer? It is easily demonstrated that the NC4000 runs faster than conventional micros. Because Forth instructions execute in a single clock cycle, the chip runs Forth code about 100 times faster than Forth running on a conventional processor. A benchmark using the Sieve of Eratosthenes reveals that the NC4000 runs Forth over 10 times faster than the 68000 runs its own machine code. (This benchmark was made on the NC4000P, the initial CMOS gate-array prototype; future revisions will increase the speed considerably.)

But these benchmarks only begin to show the potential power of the NC4000 architecture. A benchmark such as the Sieve represents a straightforward algorithm, which is best coded in the linear terms of conventional processors. There is nothing to be gained by decomposing such an algorithm into smaller modules. Although a 10-times speed improvement is nothing to be ashamed of, the Sieve benchmark doesn't let the NC4000 show its true stuff.

The real wonder of the Novix chip is that it allows execution of an elegant, high-level, modular language directly in the logic of the CPU.

Forth and the NC4000 offer three distinct benefits in terms of programming methodology. The first is the interchangeability of a word without affecting any code that invokes the word. The second is the ability to use words, which tend to be shorter and simpler, instead of subroutines which tend to be longer. The third is that more efficient subroutines encourage greater freedom in the use of modular programming. Let's examine these points more closely.

Interchangeability

In the world of software development, it's no longer enough just to get a program running. The program must also be able to be maintained and easy to change. Why would a program ever need to change, once it's running? You may need to upgrade the program to run on newer equipment. You may need to add more sophisticated features to keep up with the competition. In fact, most software groups find themselves writing families of programs; that is, they write many versions of related programs in their application field, each a variant on an earlier program.

To be maintainable, a program must be easy to read, easy to understand, and, most importantly, structurally durable. By this I mean that you can rewrite a section of the program without having the whole thing collapse.

Here is an example of how Forth encourages structurally durable code. Suppose we have a variable called STATUS. In Forth this is defined as

```
VARIABLE STATUS
```

One component of our application, a device driver, for example, writes into STATUS, another component reads STATUS. References to STATUS are present throughout the program.

In Forth, variables are locations that can be acted upon. For instance, in the phrase

```
STATUS   @
```

the word @ (pronounced "fetch") fetches the value of STATUS. In the phrase

```
STATUS   !
```

the word ! (pronounced "store") stores a given value into STATUS. In either phrase, the execution of the word STATUS is the same: it merely returns the address of the location of its data. This address is then used by the @ or ! operators (or by any other memory manipulation operation).

DISK/COVERS S100 MAIN/FRAMES SINGLE BOARD

8" & 5" WINCHESTER & FLOPPY

CHASSIS/USA

100 STANDARD MODELS

CUSTOM TOO!

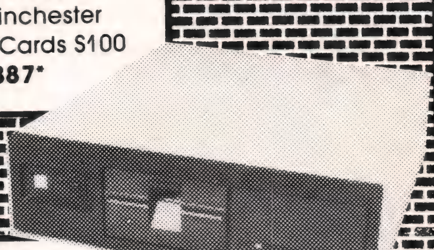
DON'T SEE WHAT YOU NEED? CALL & ASK

FROM $100 INCLUDING POWER SUPPLY

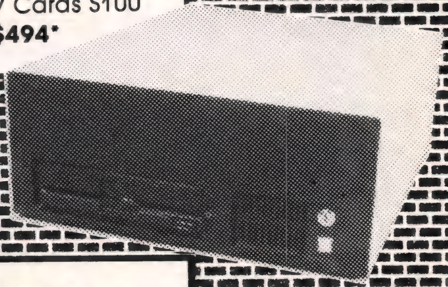
BUILT LIKE A TANK — WON'T BREAK THE BANK!

* 1 Piece: Prices lower in quantity.

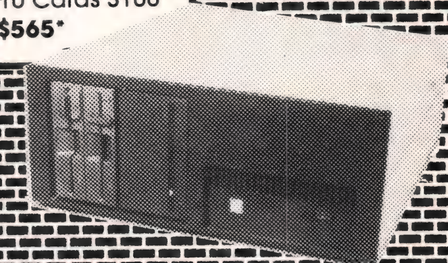
3310
5″ Floppy & Winchester
4 Cards S100
$387*



3307
8″ Floppy & 5″ Winchester
7 Cards S100
$494*



3002T
5″ Floppy & Winchester
10 Cards S100
$565*



(Disk drives and computer cards not included.)

Write or call for our brochure which includes our application note: "Making micros, better than any 'ol box computer."

INTEGRAND RESEARCH CORPORATION

8620 Roosevelt Ave./Visalia, CA 93291
209/651-1203

We accept BankAmericard/Visa and MasterCard

Circle **no. 15** on reader service card.

Now suppose we must write the next generation of the program. This time we are monitoring ten devices instead of just one and we must handle a STATUS variable for each of them. What we need is a ten-cell array and a pointer to the cell in the array for the current device. In Forth we merely define:

```
CREATE STATUSES   20 ALLOT   ( 10 cells)
VARIABLE DEVICE ( number of current device)
:  STATUS   ( -- adr)
   DEVICE @ 2*      STATUSES + ;
```

On the first line we have defined a 10-cell array (20 bytes) called STATUSES. On the second line we've created a variable that will contain the number of the current device (between 0 and 9). On the third line we've defined the word STATUS to fetch the number of the current device, multiply this by two (to compute the offset into the array in bytes), then add it to the address of the top of the array. The result is the address of the cell containing the status for the current device.

Now the other components of the application may write

```
   STATUS   @
```

and

```
   STATUS   !
```

as before, even though we have changed the definition of STATUS from that of a variable to a colon definition—from a data structure to a procedure. Forth allows us to hide the details of how STATUS is defined from the code that uses it. What appears to be a thing (a variable) to the original code is actually defined now as an action.

Words, Not Subroutines

Strictly speaking, Forth words are no different than ordinary subroutines. However, because they are optimized, and because they require no special calling command in the code that invokes them, Forth encourages a finer granularity in the decomposition of an application. In other words, Forth words tend to be much shorter and more numerous than typical subroutines in other languages. This characteristic imparts a style that is unique to Forth, and it is one of its most difficult characteristics to describe.

Here is an example of a collection of subroutines that might be written in a conventional language:

```
ENABLE-LEFT-MOTOR
ENABLE-RIGHT-MOTOR
DISABLE-LEFT-MOTOR
DISABLE-RIGHT-MOTOR
ENABLE-LEFT-SOLENOID
```

INCREMENTAL BACKUP FOR CP/M 2.2

Automatically Copy Just What's New

$40 for Qbaxl, floppy-to-floppy or hard disk with small files.

$95 for Qbax2: Splits large files, built-in catalogue, reclaims wasted floppy space.

Amanuensis, Inc.
R.D. 1, Box 236
Grindstone, PA 15442
(412) 785-2806

For CP/M 2.2 on 8" SSSD and popular 5¼" formats. Shipping $2 U.S. and Canada. $4 overseas. Qbax TM Amanuensis, Inc. CP/M* Digital Research

Circle **no. 5** on reader service card.

---

edward k. ream

Transform Your Programs with CPP—C Preprocessor Plus

Includes ALL features of the standard C preprocessor.
- Define arbitrarily complex macros with #define command.
- Include and nest files to any depth with #include command.
- Conditionally include or exclude lines with #if, #ifdef and #ifndef commands.
- Optional extra feature: Imbed formatting or other commands in your source code. (Lines starting with . or * are ignored.)

Fast and flexible
- 30 times faster than the Preprocessor published in Dr. Dobb's Journal.
- Can be used for any language, including assembler.
- Can be used as a stand-alone macro/include processor.
- Code can be used as the lexical analyzer for parsers or assemblers.

Complete
- You get complete SOURCE CODE in standard C.
- You get everything you need to use CPP immediately.
- CPP is unconditionally guaranteed. If for any reason you are not satisfied with CPP, your money will be refunded promptly.

Price: $50. Call or write today:
Edward K. Ream
1850 Summit Ave., Dept. DD
Madison, WI 53705
(608) 231-2952

TO ORDER: Specify both the operating system (MS-DOS, CP/M 80 or CPM 68K) and the disk format (8 inch CP/M or the exact type of 5¼ inch disk). Send a check or money order for $50 ($60 for foreign orders). Foreign checks must be denominated in U.S. dollars drawn on a U.S. bank. Sorry, I do NOT accept phone, credit card or COD orders. Please do NOT send purchase orders unless a check in included.

Circle **no. 13** on reader service card.

```
ENABLE-RIGHT-SOLENOID
DISABLE-LEFT-SOLENOID
DISABLE-RIGHT-SOLENOID
```

In Forth one might replace these eight subroutines with these six words:

```
ENABLE
DISABLE
LEFT
RIGHT
MOTOR
SOLENOID
```

Now our syntax reduces to simple, English words, combined in useful phrases:

```
ENABLE LEFT MOTOR
ENABLE RIGHT MOTOR
Etc.
```

Not only do we reduce the number of routines from eight to six, we also make each word's definition much shorter and simpler. The following listing shows how little code we might need to define the syntax:

```
-1 CONSTANT ENABLE ( all "one"s)
 0 CONSTANT DISABLE ( all zeros)
 1 CONSTANT LEFT (bit mask)
 2 CONSTANT RIGHT (bit mask)
 : MOTOR (action side - -) 'MOTOR SET-BIT ;
 : SOLENOID (action side - -) 'SOLENOID SET-BIT ;
```

We have defined the word SET-BIT to require three arguments: a flag, either all ones or all zeros; a bit-mask, with a "1" indicating the desired bit; and an address. SET-BIT sets the bit indicated by the mask at the given address according to the given flag. The words 'MOTOR and 'SOLENOID provide the hardware addresses of the ports that communicate to the motor and solenoid respectively; bits zero and one of these addresses control the left and right devices, respectively.

In well-written Forth code, the average definition consists of only seven words, and fits within a single line of code (or two lines if a long comment is used). Few other languages encourage you to create subroutines called LEFT and RIGHT, merely to act as "adjectives." Nor would you find routines such as MOTOR that consist of only two other words.

In conventional programming languages, subroutines tend to be longer and more complex, covering many possible options. A conventional subroutine will contain a number of conditional branches, so that it can decide how to behave depending on the desired function. Typically, many of these decisions shouldn't have to be made at run-time; they arise only because the subroutine is too gener-

Now available with 8087 Support!

MTBASIC

Basic Compiler

Features:

Multi-line functions	Multitasking
No runtime fee	Windowing
Handles interrupts	Interactive
Fast native code	Compiles in seconds

MTBASIC is easy to use since you can write programs in an interactive environment and then compile them using only one command. MTBASIC has many advanced features like multitasking, random file access, formatted I/O, assembly language calls, and ROMable code.

The MTBASIC package includes all the necessary software to run in interpreter or compiler mode, an installation program (so *any* system can use windows), demonstration programs, and a comprehensive manual.

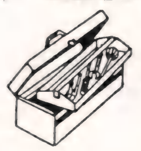
Ordering

MTBASIC is available for CP/M, MS-DOS, and PC-DOS systems for $49.95. MTBASIC with 8087 support is available for MS-DOS for $79.95. Shipping is $3.50 ($10.00 overseas). MD residents add 5% sales tax. MC, Visa, checks and COD accepted.

SOFTAID, Inc.

P.O. Box 2412 Columbia, MD 21045-1412
301/792-8096

Circle no. **88** on reader service card.



Where did AT&T and SONY find the Tools to C them thru?

The Application Programmer's Toolkit!!!

A *Wizard!Ware*™ product from Shaw ☆ American Technologies

APT™ provides you with everything you need to increase your C programming productivity, including:

- COMPLETE SOURCE CODE (over 5000 lines!)
- File handling with direct & keyed access
- Screen and Report Generators, with full screen handling for your programs
- Generic Terminal Driver for portable code
- String math functions, and string manipulation routines
- Reference Manual on Disk (over 50 pages)
- Tutorial Manual (over 25 pages) with Source for Mailing List Manager
- A host of useful Utilities, Database and File Editors
- Available for Microsoft (3.0), Lattice, Mark Williams, DeSmet, BDS, Aztec, CI-C86

C-STARTER Toolkit!: Binary APT, DeSmet C, "Programming in C on the IBM-PC"

NOW MORE AFFORDABLE!!

APT/MS-DOS versions	$395
APT/DeSmet C & BDS C versions	$295
C-Starter (binary APT, DeSmet Compiler and Book)	$295
APT/Manual only	$ 45

**** NEW PRODUCTS! Available Now or Coming Soon: ****

ADAPT: English-language, applications generator	$295
BIZ-WIZ: Comprehensive Accounting Package	$495
FLORA: Graphics Toolkit, with MAC-like capabilities	$ 95
APT-WINDOWS!: APT-Compatible/stand-alone window mgt.!	$ 95
db2c: dBaseII/III TO C Source Code Translator and Libraries!	CALL
Dr. Shaw's DOS-Shell: a UNIX-like shell for DOS	$ 95

Trademarks: MS DOS & Microsoft C/Microsoft, Lattice C/Lattice, Inc., MWC86/Mark Williams Co., DeSmet C/C Ware, CI C86/Computer Innovations, BDS C/BD Software, Aztec/Manx Software, UNIX, AT&T/AT&T, SONY/Sony Corp., Macintosh/Apple Computer, dBaseII & dBaseIII/Ashton-Tate, ADAPT/Wilmes Systems, FLORA, Biz-Wiz, WizardWare, APT, C-Starter/Shaw American Technologies.

Call (502) 583-5527

Shaw ☆ American Technologies

Wizard Ware™

830 South Second St. - Box 648
Louisville, KY 40201, USA

(C.O.D. and Foreign Orders - Add $5 Shipping/Handling)

References: Bank of Louisville, Citizens Fidelity Bank, Louisville Chamber of Commerce

Circle no. **86** on reader service card.

al. It has not been decomposed sufficiently.

The product of good decomposition is simplicity. Simplicity is the key not only to efficient, compact programs, but also to programs that are easy to write and, therefore, easy to test and debug. They are also easy to understand and, therefore, easy to maintain.

Modularization

The third way that optimized subroutines can enhance program maintainability is by allowing the program designer to break the application into very small, useful modules. Modularization is the key to well-written Forth code.

Modularization is more than merely chopping code up into small pieces. There is no way to automate good modularization. It is something that arises from the programmer's intimate understanding of the problem being solved. Allowing the problem to decompose itself naturally is a critical part of the early design phase—one that requires the programmer's experience, sensitivity, and artistry.

The best use of modularization is to decompose the application into components. A component is a resource, such as a UART driver, a queue, a linked list, a stack, etc. Each resource will typically consist of a number of functions and data structures.

The idea of component programming is to define all these functions and data structures in the same place, not scattered throughout the program as they are needed. The primary advantage is the ability to change the component if necessary by editing code in only one location.

In Forth, component programming means creating a set of names (words) to *represent* the functions and structures of each component. The set of words veil the internal algorithms and structures, the "how it works." They present only a conceptual model of the compenent described in natural language, the "what it does."

These words then become the language for describing the data structures and algorithms of components written at a higher level. The "what" of one component becomes the "how" of a higher-level component. A Forth application is nothing but a collection of components.

Thus, Forth programming consists of extending the root language toward the application, providing new commands that can be used to describe the problem at hand. Forth is an environment for creating "application-oriented languages," i.e. , programming languages designed for particular applications. Examples include robotics languages, control languages, data acquisition languages, and music languages.

Conclusion

Charles Moore, the creator of Forth and chief architect of the NC4000 processor, has claimed that the Forth chip represents "a landmark in the evolution of hardware and software. This article has only tapped the surface of the many revolutionary hardware features of the NC4000. Its high-speed computation and I/O characteristics have made the processor immediately interesting to engineers concerned with bandwidth in applications such as telecommunications and signal processing.

Its virtues are not limited to hardware features and real-time applications, however. I hope that this article has at least suggested some of the elegance of the processor's native language, and that you will have a chance to experience its unique style on your own.

Note

For more information on threaded code languages, refer to: "An Architectural Trail to Threaded-Code Systems," by Peter M. Kogge, in *Computer*, March 1982; also to *Threaded Interpretive Languages* by Ronald Loeliger, McGraw-Hill/Byte books. For further reading on the Novix processor, see "Fast processor chip takes its intructions directly from Forth," *Electronic Design*, March 21, 1985, or *Programmer's Introduction to the Novix NC4000P Microprocessor* available from Novix, 10590 N. Tantau Ave. , Cupertino, CA 95014. For a discussion of Forth methodology, see *Thinking Forth* by Leo Brodie (Prentice-Hall, 1984).

DDJ

Reader Ballot
Vote for your favorite feature/article.
Circle Reader Service **No. 192**.

8 MHZ Z-80 COPROCESSOR

TURBOSLAVE - P.C.

- 8 MHZ Z-80H
- 128K RAM
- 2 RS-232 PORTS
- MULTIPROCESSOR ARCHITECTURE
- P.C. COMPATIBLE
- FASTEST CPM COPROCESSOR
- 16 TURBOSLAVES PER P.C.
- CPM-80 COMPATIBLE
- TRUE MULTI-USER
- LOW COST - $495. RETAIL

TURBOSLAVE P.C. TURNS YOUR I.B.M. INTO A MULTI-USER MULTI-PROCESSOR SUPER MICRO. ONE TURBOSLAVE CAN BE USED AS A SIMPLE CPM COPROCESSOR. BY ADDING THE TURBODOS OPERATING SYSTEM, UP TO 16 TURBOSLAVE P.C.'S CAN BE ADDED, EACH SUPPORTING ONE USER ON A STANDARD TERMINAL. ALL CPM 2.2 PROGRAMS AND MOST MULTI-USER MPM APPLICATIONS CAN BE RUN UNDER TURBODOS. TRUE RECORD LOCKING AND PRINT SPOOLING ARE FULLY SUPPORTED.

EARTH COMPUTERS
"Building Blocks For The Super Micro"

TELEX: 9109976120 EARTH FV
P.O. BOX 8067 • DEPT. D1 • FOUNTAIN VALLEY, CA 92728
CALL: (714) 964-5784

★ S-100 VERSION AVAILABLE ★

REGISTERED TRADEMARKS: Z-80H, ZILOG INC.; TURBODOS, SOFTWARE 2000, INC.

Circle **no. 53** on reader service card.

C Programmers:

Consider 104 Ways To Be More Productive

If you find and choose the right development software, you can: cut development effort, make impractical projects feasible, and eliminate unproductive, frustrating aspects of programming.

Confused? We'll help you sort thru the huge number of alternatives. Call for comparisons or information.

We carry 27 C Compilers, 4 C Intepreters, 49 Support Libraries, 5 C source debuggers, and 19 other C Add-ons for programming with MSDOS, Macintosh, or CP/M — more than 104 products, really. Here are some of the best products available:

Learn C Programming Only $125

"Introducing C" Interpreter

Computer Innovations has done it again! This interactive implementation is combined with a full screen editor and a thorough, self-paced manual.

You can develop programs faster by getting immediate feedback. Programs will start instantly upon your command. There is no need to wait for "compile and link."

Introducing C includes demo programs, powerful C language interpreter, complete C function library, full screen editor, color graphics, and C language compatibility. PCDOS **$125**

Inventive Programming Becomes Possible with 300 + ESSENTIAL, tested, fast, rountines to Rely On.

C Utilities Library by Essential Software

Recent Enhancements to Graphics, Windows, AT Support

Every application you write is likely to require functions were you feel like you are "reinventing". Don't. Even if you use only 5% of this library, you will come out ahead on schedule and cost.

Full business Graphics, Window support, polled Communications, and Data Entry support have recently been added/upgraded along with more functions for DOS Interface and AT support. String handling, screen control, "word processor" functions, memory management, directory and path access, date handling, program chaining, keyboard and printer control are traditional strengths.

Full source code is included. No royalties are charged to include functions in your programs. 95% are C for portability and to make it practical for you to understand or modify them.

Lattice, Microsoft, C86, Mark Williams, Aztec, Desmet and Wizard C are supported. Specify which you need.

Substantial time, effort, testing and attention has been invested by Essential Software developing, documenting and supporting this comprehensive library. Make new projects practical and interesting. Use this tested and reliable library.

Some functions are PC-specific. Most support any MSDOS **$149**

SORT/MERGE Files for Clean, Fast Maintenance

with OPT-TECH SORT

Performance should not suffer with DOS or other "free" sorts. ISAMs alone are slow when 10% or even less is changed/added.

OPT-TECH includes:
- CALLable and Standalone use
- C, ASM, BAS, PAS, FTN, COBOL
- Variable and fixed length
- 1 to 9 fields to sort/merge
- Autoselect of RAM or disk
- Options: dBASE, BTrieve files
- 1 to 10 files Input
- No software max for # Records
- All common field types
- By pass headers, limit sort
- Inplace sort option
- Output = Record or keys

Try what you're using on an XT: 1,000 128 byte records, 10 byte key in 33 seconds. MSDOS. **$85**

Which Compiler Features Do You Need?

Optimizing C86 Compiler

Over the years the Optimizing C86 has evolved to be the most complete set of C compiler tools. It includes utilities, a rich library, and thorough tech support. In line 8087/287 routines run up to 100 times faster than the 8086 math package. The source code to all routines is included, so you have complete control over how they work. Thorough ROM support, Intel UDI & VMS cross versions are available.

More of the features you want include:
- special IBM-PC library • 2 math and 2 I/O libraries
- full memory utilization of the 8086/88/186/286
- compatibility with most commercial libraries
- Version 2.3 has support for source level debuggers MSDOS **$319**

Fast File Access with Source

C-Index +

C-Index + contains a high performance ISAM, balanced B + Tree indexing system with *source* and *variable length* fields. The result is a complete data storage system to eliminate tedious programming and add efficient performance to your programs.

Features include random and sequential data access, virtual memory buffering, and multiple key indexes.

With *no royalties* for programs you distribute, full source code, and variable length fields C-Index + fits what you are likely to need.

Save time and enhance your programs with C-Index + MSDOS **$359**

File Management: MultiUser/MultiLanguage

BTRIEVE

Billions and billions of bytes! That's what you can control with Btrieve's file manager. Btrieve gives you the ISAM capability you need without the maintenance headaches.

Using b-trees for optimum performance, Btrieve automatically maintains your files in sorted order on up to 24 different fields. And Btrieve offers you the fastest search algorithm available, to give you instantaneous access to any individual record. You can locate any record in 4 disk reads or less (thanks to Btrieve's RAM cache, usually less). With Btrieve you can stop wasting your time being a file clerk and concentrate on more productive tasks.

Btrieve's other features include:
- 4 gigabyte file size
- 4090 byte record length
- 255 byte key length
- duplicate, modifiable, and null keys
- up to 24 key indexes per file
- automatic file recovery after power failure

Btrieve's Local Area Network version lets you migrate your software to multiuser environments without changing your code. And offers you multiuser update capability beyond simple file locking schemes. Available *for all programming languages* as well as C. MSDOS.Single user **$199** Multiuser $595.

Btrieve. Don't settle for less.

Call for details, comparisons, or for our "C Extras Packet" with over 50 pages of information about C support products.

THE PROGRAMMER'S SHOP

The programmer's complete source for software, services and answers

128-LC Rockland Street, Hanover, MA 02339 (617) 826-7531 (800) 421-8006

Ask about COD and PO's. All formats available. Prices subject to change. Names of products and companies are generally their trademarks.

Circle **no. 99** on reader service card.

Design of a Forth Target Compiler

by Howard H. Robinson, Philip D. Morse II and Sidney A. Bowhill

Forth is a programming language that is both flexible and extensible. It also generates threaded code that is extremely compact and executes rapidly. Forth also allows the programmer to stay within a higher-level programming environment, while at the same time making use of all of the low-level capabilities of the computer. These are ideal characteristics for a language to be used in the development of programs for machines with their code in PROM.

We had been developing code for PROM-based applications for several years. Primarily we used assemblers. After we became acquainted with FIG-Forth, we realized that it had the potential to be an excellent programming environment for the development of code for PROM applications. There were, however, two serious drawbacks. Firstly, there are large sections of the compiled threaded Forth code that are not needed for the functioning of the PROM-based application, but are difficult to exclude from the code (such as the Forth compiler itself). Secondly, sections of the code must be in RAM (such as self-modifying code, variables and pseudo-registers).

What was needed to rectify these problems was a post-compiler. A post-compiler, or recompiler, selects from the compiled threaded Forth code just those segments that are necessary for the execution of the application program. It also directs any code that must be resident in RAM to the RAM area of the target machine.

By recompiling Forth programs, you can develop tight, PROMable code that executes at the top speed the target microprocessor will allow.

H. Robinson et al., *University of Illinois, 1406 W. Green St., Urbana, IL 61801*

The General Design of the CICFIG Recompiler

The program described in this article is a recompiler of the type specified above. It allows programs to be developed in Forth on a host machine and recompiled to produce the target application code. It was developed by the Central Illinois Chapter of the Forth Interest Group (CICFIG) and works in conjunction with the native host Forth compiler.

To produce the machine executable target code, source code for the application program is compiled by the standard Forth compiler within the host system. The recompiler (itself already compiled into the host Forth system) selects parts of the host system along with the compiled application code to be included in the target code. The recompiler relocates all the addresses within this code to conform to the memory map of the target machine, supplied by the application programmer. The resulting minimum length code can be executed from PROM, and potentially can perform tasks at the maximum speed of the microprocessor. The code also conforms to the memory map of the target machine, and can stand alone (i.e. does not necessarily require an operating system or monitor).

One of Forth's greatest advantages is that its interpreter can execute newly compiled code. This property greatly simplifies debugging: compiled parts of the program can be tested and debugged individually with the interpreter. To extend this advantage to the process of debugging the target application, we have written the recompiler so that the target machine code can optionally include the interpreter. When the program is fully operational, the minimum code can be generated by excluding the interpreter (and the verb headers required for the interpreter to operate). In addition, once the interpreter has been excluded, the user has access only to those parts of the program allowed by the application programmer; the fact that the program is compiled Forth code should be completely transparent to the user.

Compiled Forth Code

The recompiler has as input the application program already compiled by the host Forth system. To understand what the recompiler must do, it is necessary to understand the structure of compiled FIG-Forth-model code. The model includes three parts: the start-up code, an inner interpreter, and the dictionary. The start-up code and the inner interpreter are contiguous in low memory in the host Forth system and thus can be treated as one block that must be included in the target machine code.

The start-up code initializes the registers required for the operation of the Forth inner interpreter and then jumps to the inner interpreter. The inner interpreter uses these registers to arrange the sequential execution of verbs from the dictionary.

The dictionary is a set of verbs in one linked list. Each verb defines a sequence of operations to be executed by the machine. Each verb has three parts: a header, a code field, and a parameter field. The header specifies the name of the verb (so the verb can be identified within the dictionary), a link address (so the dictionary forms one linked list of verbs), and certain special bits that are signals to the compiler of the host Forth system.

The code field contains the address of the machine executable code to which the inner interpreter will cause

How to go from UNIX to DOS without compromising your standards.

It's easy. Just get an industry standard file access method that works on both.

C-ISAM™ from RDS.

It's been the UNIX™ standard for years (used in more UNIX languages and programs than any other access method), and it's fast becoming the standard for DOS. Why?

Because of the way it works. Its B+ Tree indexing structure offers unlimited indexes. There's also automatic or manual record locking and optional transaction audit trails. Plus index compression to save disk space and cut access times.

How can we be so sure C-ISAM works so well?

We use it ourselves. It's a part of INFORMIX®, INFORMIX-SQL and File-it!™ our best selling database management programs.

For an information packet, call (415) 424-1300. Or write RDS, 2471 East Bayshore Road, Palo Alto, CA 94303.

You'll see why anything less than C-ISAM is just a compromise.

RDS™

RELATIONAL DATABASE SYSTEMS, INC.

© 1985, Relational Database Systems, Inc. UNIX is a trademark of AT&T Bell Laboratories. INFORMIX is a registered trademark and RDS, C-ISAM and File-It! are trademarks of Relational Database Systems, Inc.

Circle no. 49 on reader service card

We will do whatever it takes to make DSD86 the best debugger available for the IBM PC.

For starters, we have by far the best design, a superior base to build from. While the competition adds new "modes" for every feature, we have a pure, consistent and expandable design. While the competition forces you to accept their particular philosophy, we offer maximized flexability. If you already have a debugger or are looking for your first, look no further because you can't do any better. We invite you to compare our debugger, DSD86, with any other on the market.

■ Recursive Command Macros & Files ■
■ Bind Macros to any key ■
■ Multi-segment Symbol Support ■
■ Symbolic Register & Stack Displays ■
■ User Customizeable Screen Layout ■
■ Superior Mode-less Design ■
■ Source Window for MS Languages ■
■ User Writable Commands & Displays ■
■ Fast Screen Update ■
■ Unique Breakpointing Facilities ■
■ 30 Day Money Back Guarantee ■

Call or write for our free report on truly advanced debugging technology which explains DSD86's design and why it is superior to the debugger you are currently using.

Take the DSD challenge: secure a money back guarantee with any of our competitors. Buy both debuggers and use them for a month. Send the one you like least back for a refund.

Only $69.95!

Soft Advances
P.O. Box 49473
Austin, Texas 78765
512-478-4763
"Programming for Productivity and Profit"
Please include $4 shipping

ATTENTION!
DDT, SID and ZSID USERS!

Why are you wasting your time with outdated debugging tools when you could be using DSD80 by Soft Advances?

DSD80 is unmatched in the CP/M-80 world and even makes most 16 bit PC programmers green with envy.

■ Full Screen Display ■
■ DDT Command Compatibility ■
■ Software In-Circuit Emulator ■
■ Superior Symbol Support ■
■ Intel and Zilog Mnemonics ■
■ 8080 and Z80 Support ■
■ Single Step Subroutines ■
■ ASCII and Binary String Searching ■
■ Port Input and Output ■
■ View Disk Files ■
■ Uses only 16k of TPA ■
■ Stack Display ■
■ Highlighted Memory Display ■
■ Dual Monitor Support ■
■ User Writeable Display ■
■ 30 Day Money Back Guarantee ■

You can experience the power of DSD80 for only $125. Our money back guarantee will protect you from overloads.

Soft Advances
P.O. Box 49473
Austin, Texas 78765
512-478-4763
"Programming for Productivity and Profit"

Please include $4 for shipping. DSD is a trademark of Soft Advances; CP/M, SID & ZSID are trademarks of Digital Research. Dealer inquires invited. Ask us about DSD86 for the IBM PC.

Circle **no. 63** on reader service card.

the processor to jump when that verb is executed at run-time. The parameter field contains the information that is necessary for the function of the verb to be accomplished. This may be data, as in the case of variables (and arrays and constants), or a list of addresses of other verbs that are to be performed sequentially (e.g. a colon definition). Alternatively, the parameter field may contain machine executable code that performs the function of the verb directly. In this case, the code field contains the address of the parameter field.

Since Forth code consists largely of lists of addresses, we have devised a novel algorithm for identifying all addresses in the host Forth system that may need to be redirected for inclusion in the target machine code. The procedure calls for two complete copies of the host Forth system, including the compiled application program. These two Forth systems are identical except that one Forth system has its origin exactly 201Hex locations higher in memory than the other (i.e. one has its origin at 100H and the other at 301H). When equivalent locations in both images are compared, all numbers (pairs of bytes) differing by 201H must be addresses. This algorithm identifies all absolute addresses, whether they are embedded in machine executable code, or used as data (such as the list of code field addresses in the parameter field of a colon definition).

Forth Systems for Target Recompilation

The recompiler described here is a program written entirely in high level Forth (MVP-Forth). The recompiler is itself compiled into an existing Forth system. The application program is also compiled into the same host Forth system. The recompiler examines the compiled application code and determines all code segments necessary for the proper functioning of the application program in the target machine. A file is generated that contains a copy of those necessary code segments. All addresses in the code segments in that file have been corrected to reflect the squeezing and the relocation of the host

Forth system to generate the target code image. This file then contains the code for the application program, which can be loaded into the target machine and executed.

The code generated by the recompiler is intended to be executed on a target machine with a microprocessor that is code-compatible with the host microprocessor; thus, the recompiler does not provide the function of a cross-compiler. At present, the recompiler can only generate code to be executed on either an 8080/Z80 compatible or an 8086/88 microprocessor based target machine. Development machines for these two families of microprocessors are common and can be quite modest (for instance, only one floppy disk drive is required).

We have considered adapting other Forths to allow the use of other microprocessors (such as the 6502). Time limitations, however, have precluded this. In addition, not all Forth systems are easily adapted for recompilation of this kind. The code to be included by the recompiler in the target machine code must comply with the following three rules.

Rule #1: All non-addresses in the two compiled Forth images discussed above must be equal. Thus, data storage areas, such as arrays, must be initialized. This rule is only relaxed in the case of variables, due to the possible use of certain system variables in both the host Forth system and in the target machine code (such as PREV, USE and SEC).

Rule #2: All inter-verb relative addressing mode jumps are forbidden (intra-verb relative jumps are allowed). This rule is necessary because the algorithm for detecting addresses does not detect relative jump addresses. The versions of the target compiler available from CICFIG (see the section on Availability, page 68) have all the inter-verb relative jumps in their kernels removed.

Rule #3: Both the host and the application compiled code must not contain machine code jumps and calls from one code level verb to another code level verb. This practice, common in FIG-Forth model systems, can be eliminated by recoding certain parts of the kernel.

MVP-Forth implemented on CP/M-80 and MSDOS machines seemed to conform most nearly to all our requirements. In order to make it comply with the above stated rules, we changed the kernel of the MSDOS version of MVP-Forth to replace all relative jumps to the inner interpreter with absolute jumps. The verbs 0BRANCH, </LOOP> and <LOOP> were recoded to remove jumps to BRANCH. A relative jump in <+LOOP> to <LOOP> was changed to an absolute jump. Both <VOCABULARY79> and <VOCABULARYFIG> were changed to contain an absolute jump to DODOES. DOES> was changed to put in an absolute jump to DODOES.

The kernel of the CP/M-80 version of MVP-Forth was changed to complete the following verbs without inter-verb jumps: —, 0BRANCH, <, </LOOP>, <?TERMINAL>, <CR>, <KEY>, <LOOP>, SECREAD, and U*. Two instructions

New Release

C + C UTILITY LIBRARY = PRODUCT

• We have over <u>300</u>
<u>complete, tested, and, documented functions.</u>
All source code and demo programs are included.

• The library was specifically designed for software development on the IBM PC, XT, AT and compatibles. There are <u>*no royalties.*</u>

• Over 95% of the source code is written in C. <u>*Experienced programmers*</u> can easily "customize" functions. <u>*Novices*</u> can learn from the thorough comments.

We already have the functions you are about to write

Concentrate on software development—not writing functions.

THE C UTILITY LIBRARY includes:
• Best Screen Handling Available • Windows • Full Set of Color Graphics Functions • Better String Handling Than Basic • DOS Directory and File Management • Execute Programs, DOS Commands and Batch Files • Complete Keyboard Control • Extensive Time Date Processing • Polled ASYNC Communications • General DOS BIOS gate • Data Entry • And More •

• The Library is compatible with: Lattice, Microsoft, Computer Innovations, Mark Williams and DeSmet. Available Soon: Digital Research, Aztec and Wizard.

C Compilers: Lattice C—$349. Computer Innovations C86—$329; Mark Williams C—$449.

C UTILITY LIBRARY $185. Special prices on library & compiler packages.

Order direct or through your dealer. *Specify compiler* when ordering. Add $4.00 shipping for UPS ground, $7.00 for UPS 2-day service. NJ residents add 6% sales tax. Master Card, Visa, check or P.O.

ESI
ESSENTIAL SOFTWARE, INC
P.O. Box 1003 Maplewood, New Jersey 07040 914 762-6605

Circle no. **138** on reader service card.

were added to the beginning of the start-up code that read and load an initial value into the cell that is to function as the pseudo-register for the return-stack pointer. This was necessary to allow this cell to be initialized in RAM when the start-up code segment is in PROM.

In both versions, the verb <+LOOP> has an inter-verb jump to BRANCH. This case is handled as an exception by the recompiler. We intended that the compiler of Forth should not be included in the resultant target machine code. Attempts to use the compiler in the target machine will have unpredictable results.

User Interface

Generation of the target machine code with this recompiler is a two-step process. The two steps are performed by the two host Forth systems TARG1 and TARG2. These two Forth systems have the recompiler already compiled into them and are identical except that they have their origins 201H locations apart.

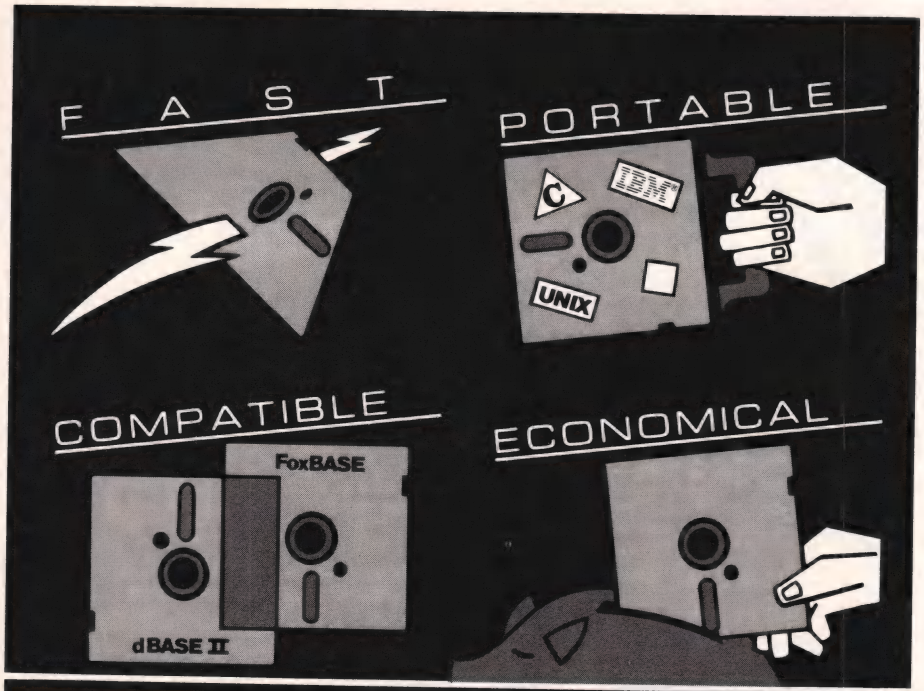
The application programmer specifies the memory map of the target machine by loading the constants indicated in Figure 1 (page 57). These constants indicate where the origin of the target machine code is and to what value pointers to any desired RAM areas of the target machine are initialized. For simplicity, this memory map conforms to the memory map of MVP-Forth. The application programmer may rearrange this map to suit the nature of the particular problem.

The application programmer must specify the verb in the host dictionary that is to serve as the cold start verb of the target machine code. This is done by loading the constant TCOLDCFA with the code field address (cfa) of the target machine cold start verb. This verb should perform any initialization of the Forth system that may be required by the application program (for example, the "user area" in RAM may need to be initialized). The examples given in the following sections explain other initializations that may be required, depending upon the nature of the application program. The cold start verb must invoke the application program. In this way the recompiler can detect and include in the target code all verbs necessary for the operation of the cold start verb and the application program.

Upon start-up of the target machine code, the start-up routine will first be performed. This will start the inner interpreter, which executes the cold start verb and then the application program.

The procedure for the generation of target code is outlined on page 76. The recompiler is initiated with a zero on the parameter stack. This signals to the recompiler that headers should be removed from all verbs to be included in the target machine code. Alternatively, if the parameter field address (pfa) of the root verb of the application is on top of the stack, then headers will be included in the target machine code. This allows verbs in the target machine code to be "found" (by -FIND). In this case the cold start verb usually invokes an interpreter such as the verb QUIT (see example #3).

In example #1 (see screens



FoxBASE.

The Breakthrough You've Been Seeking In A Database Management System.

Unsurpassed Program Development Speed.

FoxBASE™ uses a state-of-the-art B+ Tree index structure and LRU buffering scheme which greatly speed access to your data. A sophisticated virtual storage technique saves you valuable time by insuring that frequently referenced programs are retained in memory in compiled form. And automatic 8087/80287 coprocessor support gives you ultraquick speed—as much as six times the speed of dBASE II.®

Highly Portable.

Because it's written in C, FoxBASE is a highly portable interpreter/compiler. Thus, your applications need not be changed when porting from one machine or operating system to another. Only FoxBASE itself must be modified. This portability protects your investment in programs by insuring their usability in future machine and operating system environments.

dBASE II Compatible.

FoxBASE is both source language—including **full macro usage**—and data file compatible with Ashton-Tate's popular dBASE II database language. This puts thousands of public-domain and commercially available dBASE II programs at your disposal.

An Economical Investment.

For as little as $10 per license, you can distribute FoxBASE with your applications. FoxBASE even comes with a 30-day moneyback guarantee.

MS-DOS:	Development Pkg.	$395
	Runtime Pkg.	$695
AOS/VS:	Development Pkg.	$995
	Runtime Pkg.	$1995
UNIX™ and XENIX®:	(priced according to host)	

Don't be outfoxed by the others. Call or write Fox Software today.

FoxBASE™

From
FOX SOFTWARE, INC.
21475 Holiday Lane, Perrysburg, OH 43551
419 874-0162

FoxBASE is a trademark of Fox Software, Inc. dBASE II is a registered trademark of Ashton-Tate. UNIX is a trademark of Bell Laboratories. XENIX is a registered trademark of Microsoft Corp.

Circle **no. 40** on reader service card.

200–201, page 76) the recompiler is used to generate headerless code with the Forth outer interpreter excluded for a disk-based task. This program accepts HEX numbers from the keyboard and sends them to the MSDOS device PRN (usually the printer). This is useful for setting up dot-matrix printers in various modes.

Screen #200 loads a 1 into the constant MAPFILE to signal the generation of a DOS file, <codefile>.MAP, which will contain (in ASCII) the addresses and names of all verbs in the target machine code. This screen also loads the constants that define the memory map of the target machine code (see Figure 1). Loading these constants is required prior to the initiation of the recompiler.

Screen #201 contains the application program for example #1. The verb TINPUT is for numerical input from the keyboard. This avoids the Forth verb ABORT" that threads to the outer interpreter (which we are trying to exclude). LP sends a byte to the PRN device and APP is the router and root verb of the application program. Screen #201 also has the verb TCOLD which performs the initialization required to load the "user area" of RAM. The constant TCOLDCFA has been loaded with the code field address of the verb to be executed upon start-up. The code generated by this program is less than 2K.

Example #2 (screens 202–203, see page 76) demonstrates the minimum code length that can be generated by the MSDOS MVP-Forth version of the recompiler. Screen #202 loads the memory map for the target machine. Screen #203 has the verb TCOLD, which simply performs a return to the operating system. The code generated is 412 bytes, and includes the Forth start-up code, the inner interpreter, a group of mandatory verbs (which includes the run-time routine DOCOL, for instance), and INTCALL. Eleven verbs are included in the target machine code in this example.

Example #3 (screens 204–207, see page 76) demonstrates that the Forth interpreter can be included in the target machine code. The verbs in the target code must have headers so that they can be found by the outer inter-

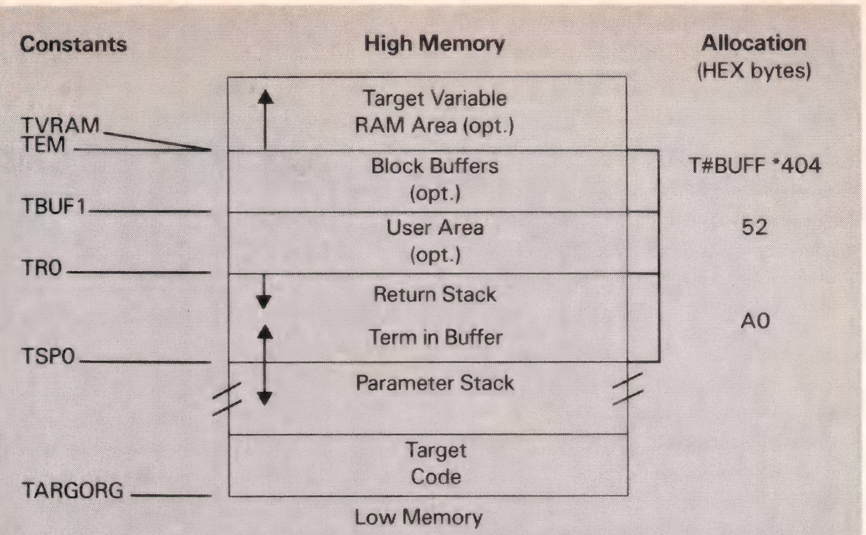


FIGURE 1
Constants Loaded by Application Programmer to Specify Memory Map

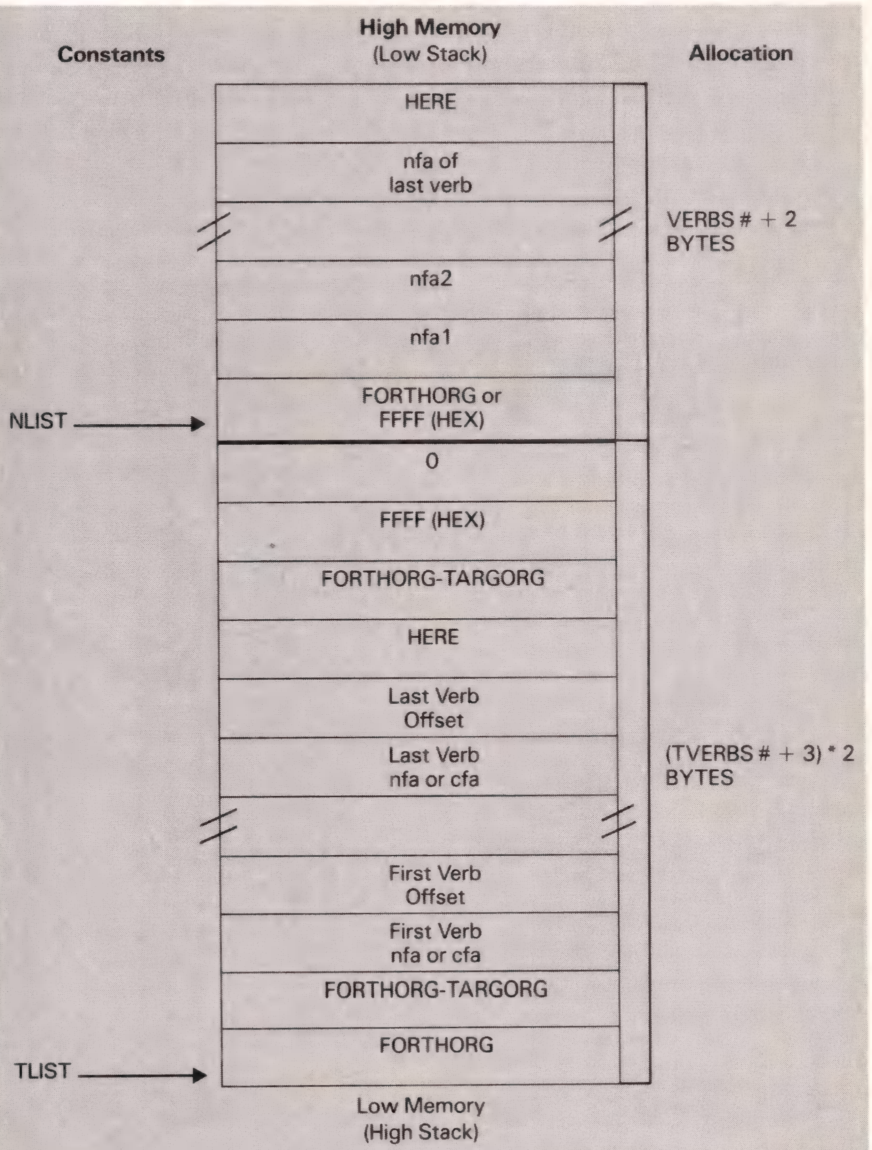


FIGURE 2
nfa and Target Offsets List

preter. Care must be taken to ensure that the Forth virtual memory block disk verbs are not included. Screens #205–206 have the Forth outer interpreter tailored to exclude references to the disk block verbs. Screen #207 has a test application program (APP) and the cold start verb TCOLD. The vocabulary link in Forth must also be initialized. The recompiler is started by typing ' APP TARGET, rather than 0 TARGET, as in a headerless case. The cfa of the root verb of the application program is on the parameter stack; knowledge of it is necessary to the recompiler since the verb TCOLD pointed to by the constant TCOLDCFA does not thread to APP.

Example #4 (screens 208–212, page 78) shows that the verbs of the Forth virtual memory disk block handlers can be included without the outer interpreter (disconnecting certain groups of routines within the host Forth system can be a bit tricky). This program reads in a number of Forth blocks (in ASCII) and writes their contents to a DOS file (in ASCII). This is done so that the file can be manipulated by your favorite word processor (such as WordStar; you must set the page length to 27 to manipulate 16 line Forth screens in document mode from the DOS file generated from this program).

In screen #208, the target memory map is loaded to indicate that 2 disk block buffers are to be allotted. Screens #209–210 set up the DOS file and transfer the information from the blocks to the DOS file. Trailing spaces are calculated by CHARS and removed by BLOCK>FOUT. Screen #211 has the verb TCOLD, which initializes the program. Several constants are loaded in TCOLD. It is important to be sure that the verbs referenced indirectly in TCOLD as literals (FIRST, LIMIT, #BUFF, and C/L) are included in the target machine code by direct references elsewhere in the program.

The Forth verb <R/W> has ABORT" in its definition. Consequently, ABORT, QUIT, etc. would all be included, although they would be quite useless, since the code generated is headerless. Thus, <R/W> should be revectored to a new verb which does not have this problem. Screen #212 has the verb <TR/W>, which is identical to the normal <R/W> except that the ABORT" construction has been removed. <TR/W> is re-vectored through 'R/W so that <TR/W> will be included by the recompiler rather than <R/W> (and the entire Forth outer interpreter).

The program in example #5 (screens 212–217, page 78) provides the inverse function of the program in example #4. In the DOS-ASCII files the <cr> <lf> pair is present at the end of each line of text, and unnecessary spaces are not present. These programs allow interchange between Forth blocks and DOS files. Example #5 removes the <cr> <lf> pairs and inserts spaces at the end of each line to fill-out each line of the Forth block. In screen #215, the verb TVARIABLE demonstrates how the RAM variable storage area can be used.

It may be necessary to generate code for a low-memory jump table to handle a start-up jump and interrupt jumps. In simple cases, this can be handled by hand patching. In more complex cases the recompiler can be instructed to generate a file (by setting the constant MAPFILE, 1 ' MAPFILE !) containing the memory map (in ASCII) of all verbs in the target machine code. The code for the low-memory jump table can be generated by a program that queries the memory map file for addresses.

Error Messages and Warnings

During the process of target compilation, if the recompiler finds numbers that are not equivalent in the two Forth images and are not identifiable as addresses (or the two bytes of the data area of a variable), then an error is indicated ("Images not matched at <address>"). A brief dump is then produced, the dictionary is "unsmudged", and control is returned to the interpreter. This error is generally the result of an uninitialized data area within a verb. The application programmer must then inspect the

A FULL C COMPILER FOR $49.95

The Eco-C88 C compiler is setting a new standard for price and performance. Compare Eco-C88's performance to compilers costing up to 10 times as much:

	Seive	Fib	Defref	Matrix
Execute	12.1 sec.	43.1 sec.	13.7 sec.	21.3 sec.
Code Size	7782	7754	7772	9120
Compile-link	76 Sec.	77 Sec.	77 Sec.	92 Sec.

Eco-C88 Rel. 2.20, on IBM PC with 2 floppy disks, 256K. Benchmarks from Feb., 1985 **Computer Language.**

Eco-C88 includes:
* All operators and data types (except bit fields)
* Error messages in English with page numbers that reference the **C Programming Guide** – a real plus if you're just getting started in C.
* Over 170 library functions, including color and transcendentals
* New Library functions for treating memory as a file
* User-selectable ASM or OBJ output (no assembler required)
* 8087 support with 8087 sensed at *runtime*
* cc and "mini-make" for easy compiles (with source)
* Fast, efficient code for all IBM-PC, XT, AT and compatibles using MSDOS 2.1 or later.
* Complete user's manual

If ordered with the compiler, the C library source code (excluding transcendentals) is $10.00 and the ISAM file handler (as published in the **C Programmer's Library,** Que Corp) in OBJ format is an additional $15.00. Please add $4.00 for shipping and handling. To order, call or write:

MasterCard VISA AMERICAN EXPRESS

Ecosoft Inc.
6413 N. College Avenue
Indianapolis, IN 46220
(317) 255-6476 • 8:30-4:30

ECOSOFT INC.

Eco-C (Ecosoft), MSDOS (Microsoft), UNIX (Bell Labs), CP/M (Digital Research), Z80 (Zilog), 8086, 8087, 8088 (Intel).

Circle **no. 35** on reader service card.

AZTEC

The C for Microcomputers

PC-DOS, MS-DOS, CP/M-86, Macintosh, Amiga, Apple II, CP/M-80, Radio Shack, Commodore, XENIX, ROM, and Cross Development systems

MS-DOS, PC-DOS, CP/M-86, XENIX, 8086/80x86 ROM

Manx Aztec C86

"A compiler that has many strengths . . . quite valuable for serious work"

Computer Language review, February 1985

Great Code: Manx Aztec C86 generates fast executing compact code. The benchmark results below are from a study conducted by Manx. The Dhrystone benchmark (CACM 10/84 27:10 p1018) measures performance for a systems software instruction mix. The results are without register variables. With register variables, Manx, Microsoft, and Mark Williams run proportionately faster, Lattice and Computer Innovations show no improvement.

	Execution Time	Code Size	Compile/ Link Time
Dhrystone Benchmark			
Manx Aztec C86 3.3	34 secs	5,760	93 secs
Microsoft C 3.0	34 secs	7,146	119 secs
Optimized C86 2.20J	53 secs	11,009	172 secs
Mark Williams 2.0	56 secs	12,980	113 secs
Lattice 2.14	89 secs	20,404	117 secs

Great Features: Manx Aztec C86 is bundled with a powerful array of well documented productivity tools, library routines and features.

Optimized C compiler	Symbolic Debugger
AS86 Macro Assembler	LN86 Overlay Linker
80186/80286 Support	Librarian
8087/80287 Sensing Lib	Profiler
Extensive UNIX Library	DOS, Screen, & Graphics Lib
Large Memory Model	Intel Object Option
Z (vi) Source Editor -c	CP/M-86 Library -c
ROM Support Package -c	INTEL HEX Utility -c
Library Source Code -c	Mixed memory models -c
MAKE, DIFF, and GREP -c	Source Debugger -c
One year of updates -c	CP/M-86 Library -c

Manx offers two commercial development systems, Aztec C86-c and Aztec C86-d. Items marked -c are special features of the Aztec C86-c system.

Aztec C86-c Commercial System	**$499**
Aztec C86-d Developer's System	**$299**
Aztec C86-p Personal System	**$199**
Aztec C86-a Apprentice System	**$49**

All systems are upgradable by paying the difference in price plus $10.

Third Party Software: There are a number of high quality support packages for Manx Aztec C86 for screen management, graphics, database management, and software development.

C-tree $395	**Greenleaf $185**
PHACT $250	**PC-lint $98**
HALO $250	**Amber Windows $59**
PRE-C $395	**Windows for C $195**
WindScreen $149	**FirsTime $295**
SunScreen $99	**C Util Lib $185**
PANEL $295	**Plink-86 $395**

MACINTOSH, AMIGA, XENIX, CP/M-68K, 68k ROM

Manx Aztec C68k

"Library handling is very flexible . . . documentation is excellent . . . the shell a pleasure to work in . . . blows away the competition for pure compile speed . . . an excellent effort."

Computer Language review, April 1985

Aztec C68k is the most widely used commercial C compiler for the Macintosh. Its quality, performance, and completeness place Manx Aztec C68k in a position beyond comparison. It is available in several upgradable versions.

Optimized C	Creates Clickable Applications
Macro Assembler	Mouse Enhanced SHELL
Overlay Linker	Easy Access to Mac Toolbox
Resource Compiler	UNIX Library Functions
Debuggers	Terminal Emulator (Source)
Librarian	Clear Detailed Documentation
Source Editor	C-Stuff Library
MacRam Disk -c	UniTools (vi,make,diff,grep) -c
Library Source -c	One Year of Updates -c

Items marked -c are available only in the Manx Aztec C86-c system. Other features are in both the Aztec C86-d and Aztec C86-c systems.

Aztec C68k-c Commercial System	**$499**
Aztec C68d-d Developer's System	**$299**
Aztec C68k-p Personal System	**$199**
C-tree database (source)	**$399**
AMIGA, CP/M-68k, 68k UNIX	**call**

Apple II, Commodore, 65xx, 65C02 ROM

Manx Aztec C65

"The AZTEC C system is one of the finest software packages I have seen"

NIBBLE review, July 1984

A vast amount of business, consumer, and educational software is implemented in Manx Aztec C65. The quality and comprehensiveness of this system is competitive with 16 bit C systems. The system includes a full optimized C compiler, 6502 assembler, linkage editor, UNIX library, screen and graphics libraries, shell, and much more. The Apple II version runs under DOS 3.3, and ProDOS, Cross versions are available.

The Aztec C65-c/128 Commodore system runs under the C128 CP/M environment and generates programs for the C64, C128, and CP/M environments. Call for prices and availability of Apprentice, Personal and Developer versions for the Commodore 64 and 128 machines.

Aztec C65-c ProDOS & DOS 3.3	**$399**
Aztec C65-d Apple DOS 3.3	**$199**
Aztec C65-p Apple Personal system	**$99**
Aztec C65-a for learning C	**$49**
Aztec C65-c/128 C64, C128, CP/M	**$399**

Distribution of Manx Aztec C

In the USA, Manx Software Systems is the sole and exclusive distributor of Aztec C. Any telephone or mail order sales other than through Manx are unauthorized.

Manx Cross Development Systems

Cross developed programs are edited, compiled, assembled, and linked on one machine (the HOST) and transferred to another machine (the TARGET) for execution. This method is useful where the target machine is slower or more limited than the HOST, Manx cross compilers are used heavily to develop software for business, consumer, scientific, industrial, research, and educational applications.

HOSTS: VAX UNIX ($3000), PDP-11 UNIX ($2000), MS-DOS ($750), CP/M ($750), MACINTOSH ($750), CP/M-68k ($750), XENIX ($750).

TARGETS: MS-DOS, CP/M-86, Macintosh, CP/M-68k, CP/M-80, TRS-80 3 & 4, Apple II, Commodore C64, 8086/80x86 ROM, 68xxx ROM, 8080/8085/Z80 ROM, 65xx ROM.

The first TARGET is included in the price of the HOST system. Additional TARGETS are $300 to $500 (non VAX) or $1000 (VAX).

Call Manx for information on cross development to the 68000, 65816, Amiga, C128, CP/M-68K, VRTX, and others.

CP/M, Radio Shack, 8080/8085/Z80 ROM

Manx Aztec CII

"I've had a lot of experience with different C compilers, but the Aztec C80 Compiler and Professional Development System is the best I've seen."

80-Micro, December, 1984, John B. Harrell III

Aztec C II-c (CP/M & ROM)	**$349**
Aztec C II-d (CP/M)	**$199**
C-tree database (source)	**$399**
Aztec C80-c (TRS-80 3 & 4)	**$299**
Aztec C80-d (TRS-80 3 & 4)	**$199**

How To Become an Aztec C User

To become an Aztec C user call 1-800-221-0440 or call 1-800-832-9273 (800-TEC WARE). In NJ or outside the USA call 201-530-7997. Orders can also be telexed to 4995812.

Payment can be by check, COD, American Express, VISA, Master Card, or Net 30 to qualified customers.

Orders can also be mailed to Manx Software Systems, Box 55, Shrewsbury, NJ 07701.

How To Get More Information

To get more information on Manx Aztec C and related products, call 1-800-221-0440, or 201-530-7997, or write to Manx Software Systems.

30 Day Guarantee

Any Manx Aztec C development system can be returned within 30 days for a refund if it fails to meet your needs. The only restrictions are that the original purchase must be directly from Manx, shipped within the USA, and the package must be in resalable condition. Returned items must be received by Manx within 30 days. A small restocking fee may be required.

Discounts

There are special discounts available to professors, students, and consultants. A discount is also available on a "trade in" basis for users of competing systems. Call for information.

MANX

To order or for information call:

800-221-0440

RUN/C:™ The C Interpreter

Only $149.95!

For both
the beginner and the C professional,
RUN/C: The C Interpreter
makes program development easier and faster. With **RUN/C** all those C programs you've been writing — or have been wanting to write — can be up and running in a fraction of the time.

The beauty of **RUN/C** is that it provides a BASIC-like user interface for C; it allows the user to edit and debug code immediately and interactively.

RUN/C is the first program to make C a user-friendly language.

Although C is structured, compact and FAST, the writing and testing of C programs is often a tedious process. **RUN/C** helps bring up to speed both your programs and your C programming skills. C programming has never been so fast and enjoyable!

When running under **RUN/C**, your C program performs exactly as it would if it were compiled (although slower since **RUN/C** is a *true* interpreter). If your program does have an error, **RUN/C** finds it, gives you a comprehensive error message and allows you to correct the error on the spot. Once you are completely satisfied with your C program it can be SAVEd, then compiled and linked using your favorite C compiler.

RUN/C offers easy and familiar commands such as LOAD, LIST, SAVE, RUN, etc. A powerful line editor is built right in. **RUN/C's** SHELL command will also allow you to use your own editor for extensive full-screen editing, and then return your newly edited program to **RUN/C** — all within a single, unified environment.

RUN/C offers:

- A robust implementation of standard Kernighan and Ritchie C.
- Full floating point, 8087 math chip support, **structures, unions, initializers,** casts and more than 100 built-in standard C library functions.
- An easy-to-read 475-page manual filled with useful examples to help you master the C language.
- TRON, TRACE and DUMP diagnostics PLUS a program profiler.
- Printer and asynchronous communications support.
- A full set of buffered and unbuffered file I/O functions.
- Nearly 100 sample C programs on disk illustrating the most important C functions and concepts.
- System Requirements: IBM® PC or compatible with PC-DOS 2.0 or MS™-DOS 2.0

CALL for information on non-IBM compatible MS-DOS systems.

For immediate delivery or more information:

Call
1-800-847-7078
In NY, 1-212-860-0300

Lifeboat™

Lifeboat™ Associates 1651 Third Ave. New York, NY 10128

RUN/C is a trademark of Age of Reason Co.

Circle no. 125 on reader service card.

dump to determine the offending verb and recode or initialize the verb. After repairing the problem, the target compilation process must be re-initiated from the beginning.

A temporary DOS file (TARGET.TMP) is generated on the default drive by TARG1 to hold an image of itself. TARG2 expects to find this file on the default drive. If this file is missing, TARG2 requests that it be replaced ("TARGET.TMP not found on default drive.").

A warning ("<address> <ID.> —Code verb assumed.") is issued by both TARG1 and TARG2 if there is a verb with an address in its code field that the recompiler cannot recognize. The offending verb will be included in the target machine code and the recompiler will continue.

The most common problem we have encountered while using this target compiler has been caused by a verb missing from the application code that seemingly should have been included. This situation arises because the missing verb is referenced only indirectly (as a literal for instance). That is, the missing verb's pfa is compiled into a verb as a literal by the verb ' at compile-time. If all references to a verb are of this type then the recompiler will not include this verb in the application code. The recompiler was designed this way because there is no way to anticipate the intention of the application programmer for the literals compiled into the definition of a verb.

If the application program crashes, the programmer should first inspect the source code to see if there are any missing verbs. This can be done by checking all verbs referenced by literals (from ') against the verbs included in the application code listed in the DOS application code map file. If a verb is missing, then the application code must be changed to force inclusion of the missing verb by explicitly referencing that verb.

It is our intention that verbs not required for the run-time activity of the application program not be included in the target machine code. We have found that the outer interpreter verbs or the disk block virtual memory verbs are sometimes included in the code when they have no intended use in the application. The most common source of this problem is run-time error handling of Forth through the verb <ABORT">. This verb leads the recompiler far into the outer interpreter and BLOCK verbs of the host Forth system. To avoid including unwanted verbs, the application programmer needs to identify an unwanted verb (e.g. QUIT or R/W etc.). A program has been devised (to be described elsewhere) to determine the threads that lead to the unwanted verbs. The application programmer can redefine the particular verb that leads to the unwanted verbs. Example #3 demonstrates this situation. In this example some of the verbs of the interpreter have been redefined to exclude any reference to the disk block virtual memory verbs. This is quite a general procedure and demonstrates the great flexibility of Forth in allowing procedures that are central to the functioning of the host system to be redefined for inclusion in the target code.

Inside the Target Compiler

The verb TARGET sequentially invokes the major tasks required by the recompilation process. The first of these tasks, APPTAG, tags (sets bit 5 of the header, the SMUDGE bit) all verbs in the host that are to be included in the application code.

The constant HEADERS, if it is non-zero (that is, the cfa of the application root verb), signals that all verbs necessary for the functioning of both the application root verb and the cold start verb (referenced by the constant TCOLDCFA) are to be tagged. In this case, headers will be included in the target code. The cold start routine must lead to an interpreter (such as QUIT).

If the constant HEADERS is zero, then only those verbs needed for the functioning of the cold start verb are tagged and the cold start verb must thread to the application program. In this case, headers will not be included in the target code. If the interpreter or other unwanted verbs are unintentionally tagged, the source code must be modified to exclude references to unwanted verbs.

To search the tree structure defined by the application program, the recursive verb TREETAG is initiated with the cfa of the cold start of the application on the stack. TREETAG determines the type of the verb being examined at that node in the tree. Variables, constants, and machine-code verbs are simply tagged and TREETAG terminates at that level of recursion. If TREETAG is examining a colon verb, then each entry in the parameter field is the cfa of another verb that must be searched. TREETAG puts a cfa on the stack and then calls itself. This process is repeated for all entries in the parameter field. The verb PFSKIPS handles entries in the parameter field that are exceptions (entries that are to be skipped, such as

COHERENT
for the AT

At last –
a UNIX compatible O/S with elaborate hard disk recovery routines!

- COHERENT is the best UNIX compatible O/S for networking, process control and multiuser applications with highly efficient code generation, configurable kernel and flexible disk partitioning.
- Support available for high resolution graphics, image capture, mouse and other hardware.
- A complete COHERENT development environment that includes a screen editor and source code control system.
- Total object code compatibility between the IBM AT, XT, and PC.
- Available with UNIX System V compatible support for interprocess communication enhancements and high speed multiline communication cards (up to 38.4 Kbaud).
- Can be easily programmed into a turnkey environment. Flexible turnkey and runtime licensing available.
- Professional technical support provided.

To discuss your system requirements call: (604) 294-6201

INETCO Systems Ltd.
2457 Beta Ave., Burnaby, B.C., Canada V5C 5N1

COHERENT is a trademark of Mark Williams Co. UNIX is a trademark of Bell Laboratories. IBM AT, XT and PC are trademarks of International Business Machines Corp.

Circle no. 2 on reader service card.

DE SMET C
8086/8088 Development Package
$109

FULL DEVELOPMENT PACKAGE
- Full K&R C Compiler
- Assembler, Linker & Librarian
- Full Screen Editor
- Execution Profiler
- Complete STDIO Library (>120 Func)

Automatic DOS 1.X/2.X SUPPORT
BOTH 8087 & S/W FLOATING POINT
OVERLAYS
OUTSTANDING PERFORMANCE
- First and Second in AUG '83 BYTE benchmarks

SYMBOLIC DEBUGGER $50
- Examine & change variables by name using C expressions
- Flip between debug and display screen
- Display C source during execution
- Set multiple breakpoints by function or line number

DOS LINK SUPPORT $35
- Converts DeSmet.O to DOS.OBJ Format
- LINKs with DOS ASM
- Uses Lattice® naming conventions

C WARE CORPORATION
P.O. Box C, Sunnyvale, CA 94087
(408) 720-9696
Street Address: 505 W. Olive, #767 (94086) Call for hrs.

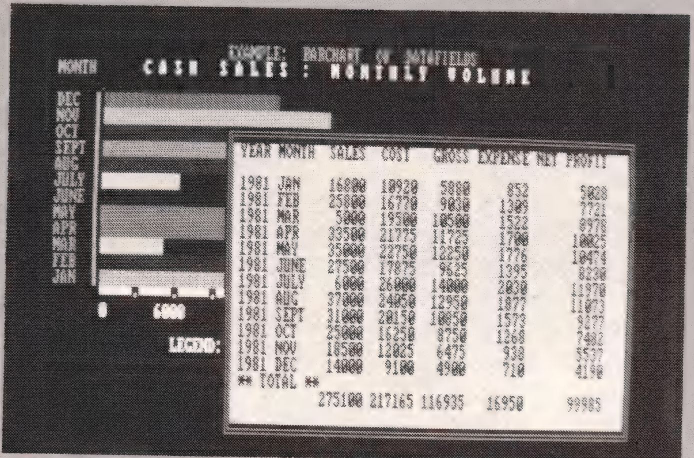
All orders shipped UPS surface on IBM format disks. Shipping included in price. California residents add sales tax. Canada shipping add $5, elsewhere add $15. Checks must be on U.S. Bank and in U.S. Dollars. Call 9am–1pm to CHARGE by VISA/MC/AMEX.
Foreign Distributors: AFRICA, HI-TECH SVCS, Gaborone 4540 or Telex 2205BD LANGER • ENGLAND: MLH Tech, 0606-891146 • JAPAN: JSE 03-486-7151 • SWEDEN: ESCORT DATA 08-87 41 48 or THESEUS KONSULT 08-23 61 60

Circle no. 57 on reader service card.

dWINDOW©!

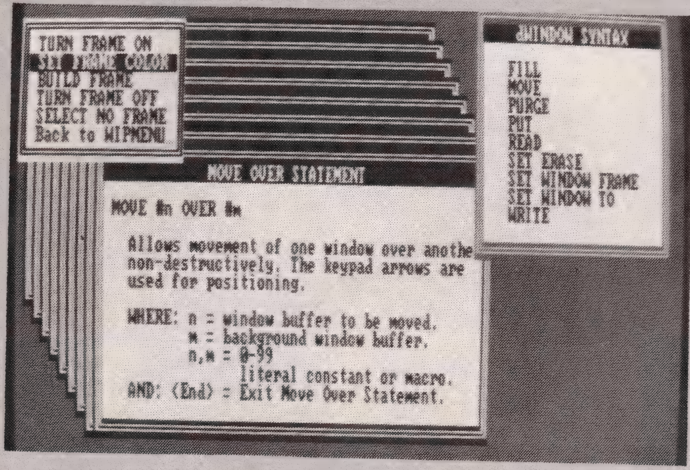
An assembly language (9K) program adding **25 NEW WINDOWING SYNTAX STATEMENTS** to dBASE II, featuring:

- In-memory and disk-based window files
- Speeds up application program execution time
- Fast video response time
- Reduces application code requirements
- Fully compatible with dBASE II .PRG files
- Full screen, free text input with TYPE command



```
          EXAMPLE:  BARCHART OF DATAFIELDS
MONTH   CASH SALES : MONTHLY VOLUME

DEC
NOV
OCT
SEPT         YEAR MONTH  SALES  COST   GROSS EXPENSE NET PROFIT
AUG
JULY         1981 JAN    16800  10920   5880    852      5028
JUNE         1981 FEB    25800  16770   9030   1309      7721
MAY          1981 MAR     5000  19500  10500   1522      8978
APR          1981 APR    33500  21775  11725   1700     10025
MAR          1981 MAY    35000  22750  12250   1776     10474
FEB          1981 JUNE   27500  17875   9625   1395      8230
JAN          1981 JULY    6000  26000  14000   2030     11970
             1981 AUG    37000  24050  12950   1877     11073
 0     6000  1981 SEPT   31000  20150  10850   1573      9277
             1981 OCT    25000  16250   8750   1268      7482
     LEGEND: 1981 NOV    18500  12025   6475    938      5537
             1981 DEC    14000   9100   4900    710      4190
             ** TOTAL **
                        275100 217165 116935  16950     99985
```

- Character Graphics / Modem compatible
- User Friendly MENU GENERATION (WIP.PRG)



```
TURN FRAME ON
SET FRAME COLOR
BUILD FRAME
TURN FRAME OFF
SELECT NO FRAME
Back to WIPMENU

                 MOVE OVER STATEMENT
MOVE #n OVER #m

  Allows movement of one window over anothe
  non-destructively. The keypad arrows are
  used for positioning.

  WHERE: n = window buffer to be moved.
         m = background window buffer.
         n,m = 0-99
               literal constant or macro.
  AND: (End) = Exit Move Over Statement.
```

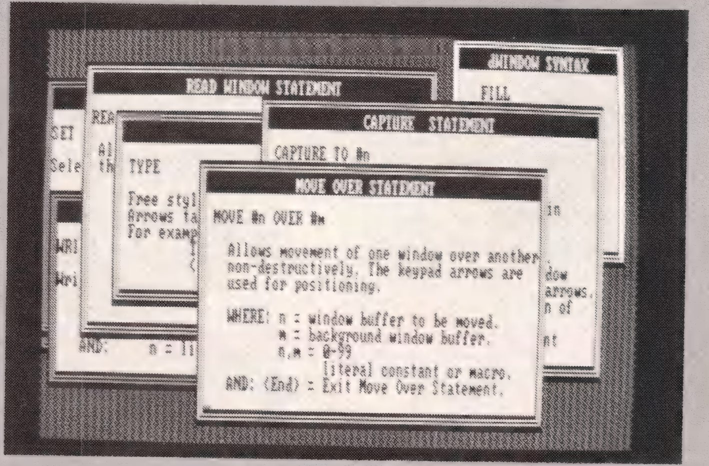
```
dWINDOW SYNTAX

FILL
MOVE
PURGE
PUT
READ
SET ERASE
SET WINDOW FRAME
SET WINDOW TO
WRITE
```

- READ WINDOW menu selection available
- Executable DOS command at dot prompt or under .PRG control
- INKEY statement allows keyboard entry of full ASCII set
- All 256 characters and color attributes selectable for both mono and color systems. (Bit-mapped graphics card or mouse not necessary.)
- PRINT SCREEN statement
- CAPTURE – full or partial screen to window
- SET WINDOW PATH – disk-based, tree-structured window files
- SET SYNC on/off – VIDEO adjustment of snow and glitches
- 100 dynamically allocated memory window buffers allowable

- Overlapping, moving windows
- Read current CURSOR position
- Select ERASE attribute (All shades and colors)
- User defined windows; WINDOW STRUCTURE programmable
- Change background / foreground colors easily

Improvements to dBASE II:

- Line 0 made available for @ SAY or GET statements
- Line 25 available for @ SAY statements
- Split-Screen MODIFY COMMAND
- *** SYNTAX ERROR *** processing compatibility



```
READ WINDOW STATEMENT
CAPTURE STATEMENT
CAPTURE TO #n
dWINDOW SYNTAX
FILL
TYPE
Free styl
Arrows ta
For examp

                 MOVE OVER STATEMENT
MOVE #n OVER #m

  Allows movement of one window over another
  non-destructively. The keypad arrows are
  used for positioning.

  WHERE: n = window buffer to be moved.
         m = background window buffer.
         n,m = 0-99
               literal constant or macro.
  AND: (End) = Exit Move Over Statement.
```

Training and Learning tools:

- on-line, menu-driven HELP
- 66 Window File examples
- 33 dBASE II .PRG files — sample applications using dWINDOW syntax statements and algorithms, including:
  - The IDEA . . . A menu-driven guide to dWINDOW
  - WIP . . . WINDOW INFORMATION PROCESSING and GENERATION
  - BARCHART . . . Columnar Report Form with Barcharts
  - NOTEPAD . . . On-line database of Quick Notes
  - ZOOM . . . Automatic program generation; scaled windows
  - PAINT – Graphic art controlled with keypad arrows

System Requirements:

- IBM PC, XT, or AT (and compatibles) with 192K
- DOS version 2.0
- MONOCHROME or Color display
- Assembler Language
- dBASE II version 2.4 or 2.41

Price: $165.00 plus $4.00 (shipping)

To order, phone (503) 221-1806 or write:

Liberty Bell Publishing 4640 S.W. Macadam, Suite 150 Portland, Oregon 97201

Circle **no. 60** on reader service card.

branches and character strings).

Verbs that are "CREATE DOES>" defining verbs are tagged by having VERBTAG find their name field addresses with the verb MAXNFA. MAXNFA recursively searches the entire host dictionary (when initiated properly; see the glossary, page 69) and returns the maximum nfa less than the address of the DOES> part of the "CREATE DOES>" construction. Further searching of the DOES> portion of the defining verb is performed by the verb CREATE-DOES>. This verb functions similarly to TREETAG and recursively calls TREETAG by the forward-reference-executing verb >TTAG. At present, the CREATE portion of the CREATE-DOES> construction will be included (wastefully) in the target code; however, no searching or tagging of the verbs referenced in this portion is performed. The exact form of "CREATE DOES>" defining verbs varies with different Forth implementations; some adjustments to the recompiler may be necessary with other FIG-Forth model systems.

APPTAG directs TFIX;CODE and TFIXNULL to tag certain verbs that are required in the target code; also the root verb of the application is searched if headers are requested.

VERBLISTS directs the creation of two lists on the parameter stack. The first list (proceeding down in memory) is a numerically sorted list of all name field addresses in the host dictionary. This list reflects the physical order of the verbs in memory rather than the order defined by the links within headers of the verbs. This physical ordering is useful during the elimination of unwanted verbs. The second verb list contains pairs of numbers. There is one pair for each tagged verb in the host. The first number is the address of the beginning of the memory segment for that tagged verb in the host system. This is either a cfa or an nfa, depending on whether or not headers are to be included in the target code. The second number of each pair is the numerical difference between the first number of the pair and the address of the start of that verb in the target machine code. This information will be used to recalculate all of the addresses in the target code to reflect the squeezing out of the unnecessary verbs from the host Forth system.

The verb FRAMEIN sets aside the space (see Figure 2, page 57) on the stack for the nfa list and the target offsets list. The size of this space is determined by the verb COUNTVERBS which recursively searches all vocabulary branches of the host dictionary, determining the total number of verbs in the host and how many of those verbs are tagged (the variables VERBS# and TVERBS#). FRAMEIN initializes certain pointers to these lists and also initializes the beginning and ending entries that are used as signals for the list searching verbs.

The verb OFFSETS enters into the target offsets list the starting address of the verbs to be in the target code. A running count of the verb locations to be in the target code is calculated. This number is used to calculate the difference between the host address and the target code address of each verb. For each verb that will be in the target, the host code address, paired with the offset (the difference calculated above), is entered into the tar-

Leap MS-DOS Software With A Single File!!

It's

SuperBatch™

More powerful than Prokey™ or Superkey™!
Stops integration nightmares with its bare bits!

SuperBatch™ is a powerful utility program that creates batch files that drive almost any program (LOTUS 1-2-3™, Wordstar™, etc.). It also continues execution from program to program. Common uses include:

- Integration of individual programs to form a system.
- Creation of self running demo systems.
- Inserting repetitive procedures into a SuperBatch™ file for simple execution.

Please send [ ] Info [ ] SuperBatch
Name ____________________
Addr ____________________
Addr ____________________
Addr ____________________
City ____________________
State ________ Zip ________ Tele ________

Please mail or call 313-645-5280 to place your order.
$79.95 with a 30 day Money Back Guarantee
Canadian orders add $10.00 for shipping.

_____ copies x 79.95 ________
MI res. add 3.20 sales tax per ________
Total ________

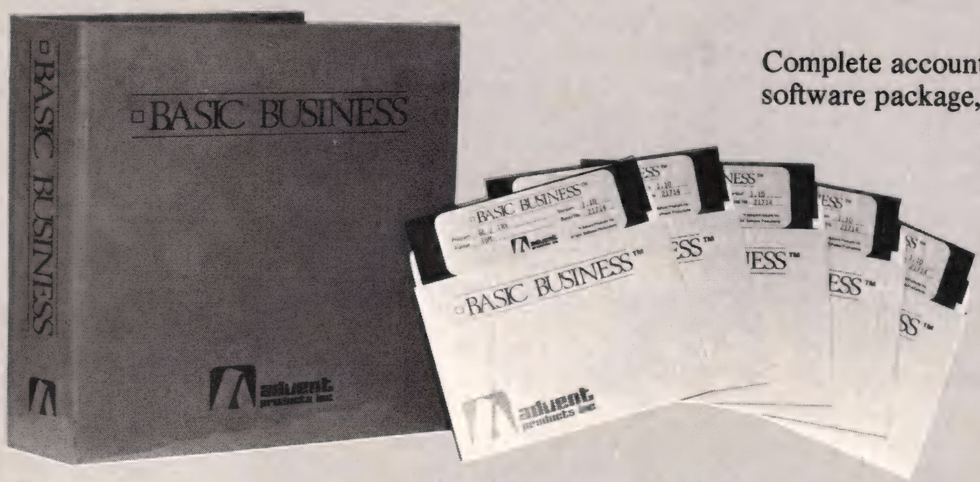
Merrill Street Software
Division of BCC Inc.
251 Merrill Street
Birmingham, MI 48011

Payment [ ] Visa [ ] MC [ ] Chk [ ] PO [ ] Bank Draft [ ] COD
Name Cr Card ____________________
Cr Card No. ____________________ Card Exp. ________

Prokey™ is a registered trademark of RoseSoft, Inc. Superkey™ is a registered trademark of Borland International. Lotus 1-2-3™ is a registered trademark of Lotus Development Corp. Wordstar™ is a registered trademark of Micropro International. SuperBatch™ is a registered trademark of BCC, Inc. SuperBatch™ requires MS-DOS 2.0 or greater.

Circle **no. 29** on reader service card.

TIMELY REPORTS, ACCURATE RECORDS, RELIABLE INFORMATION......



Complete accounting software package, only **$89.95**

......this is ◻ BASIC BUSINESS

Price is nice - and so is a long list of features - but when you choose a software package to automate your accounting, don't lose sight of the basics of good business record keeping. Basic Business, an all-in-one accounting software package, offers an excellent price and extensive feature list without sacrificing these basic business principles:

Full Audit Trails - to give you complete confidence in the accuracy of your data and provide crucial backup information when you need it. And double entry accounting keeps your books in balance.

Complete Integration - provides efficient processing for all your business transactions, updating all of your accounting records instantaneously. You only enter the transaction once, and all supporting modules are updated automatically.

Flexibility - Basic Business can be adapted to your way of doing business, including balance forward or open item accounts receivable, departmental or consolidated general ledger, variable aging periods and easy entry of manually written checks and voids.

Extensive Reporting - each accounting module provides complete reports, including master lists, transactions, journals, statements and forms. In addition, all data files are compatible with Ashton-Tate's dBase II and dBase III, for the ultimate in custom reporting capability.

With some accounting software, even packages costing hundreds of dollars more, you must make detailed estimates and complex calculations for the maximum number of customers, transactions, inventory items, etc., before using the system. Then, when your business grows to exceed these original estimates, you must repeat this process.

Not with Basic Business! All data files are automatically initialized when you install the system on your computer. Files can grow dynamically as your business increases and are usually limited only by the amount of disk storage space available (Max. 65,000 records per file). No need for cumbersome reformatting once you exceed your original estimates.

FORMS ARE NO PROBLEM! Basic Business uses standard forms for invoices, statements, checks, purchase orders, etc., which may be ordered with your company name, adddress and logo imprinted.

Basic Business is one of a family of accounting software packages, and has sold previously for several hundred dollars per module. It has been improved, updated and re-packaged to sell at a market-busting $89.95 for all seven modules; General Ledger, Accounts Receivable, Accounts Payable, Payroll, Inventory Control, Sales Order Processing and Purchase Order Processing. A Point-of-Purchase module, which controls an electronic cash drawer and allows direct entry of transactions from your sales counter, is sold separately for $99.95.

SOURCE CODE AVAILABLE

Do you have a distinct accounting problem that off-the-shelf software won't handle? Special forms or statements? Don't write your own accounting system from the ground up - start with Basic Business. Call (714) 630-0446 for all the details on source code licensing.

Basic Business can go to work for you today and is available for all popular MS-DOS (IBM and compatibles) and CP/M-80 personal computers. Compare our price, features and attention to detail. There is only one choice . . . it's Basic Business.

Basic Business	$89.95
Point-of-Purchase module	$99.95
dBase II/III file formats	$19.95

Please add $2.50 for shipping.

Minimum Hardware Required for MS-DOS computers: Color or monochrome display, 256K memory, two DS/DD floppy disk drives (or 1 floppy disk drive and hard disk recommended for Sales and Purchase Order Processing), printer w/132 columns, MS-DOS (or PC-DOS) version 2.0 or later.

Minimum Hardware Required for CP/M-80 computers: 80 x 24 character display terminal, 64K memory, two 360K disk drives (hard disk recommended for Sales and Purchase Order Processing), printer w/132 columns, CP/M-80 version 2.0 or later.

Circle **no. 4** on reader service card.

SERVICES

File Transfer Service: Advent provides a service beyond the ability of any format conversion software! We can transfer files between MS-DOS/PC-DOS, CP/M and other operating systems in 300 different 3 1/2", 5 1/4" and 8" formats. Includes Apple and Mac, Apricot, Data General One, Kaypro 2000, Eagle, Epson QX-10 & PX-8 (ROM), HP-150, and North Star computers.

ENGINEERING SOFTWARE

ACNAP: A stand-alone Electronics Circuit Analysis Program for use with passive and active circuits consisting of resistors, capacitors, inductors, transistors, op-amps, FETs, etc. Features menu driven and very fast processing times with circuits saved to disk for later use or editing.
ACNAP (CP/M & MS-DOS) $69.95

DCNAP: Stand-alone DC circuit analysis program for use with passive and active circuits containing resistors, voltage sources, independent and dependent current sources. Fast, menu- driven program with circuit saved to disk for later use or editing.
DCNAP (CP/M & MS-DOS) $69.95

Plotpro: Scientific graph printing program. Prints on 80 or 132 column printer. Create linear, semi-logarithmic, and full logarithmic plots with one or two Y axes in auto or forced scale.
Plotpro (CP/M & MS-DOS) $69.95

SPP: This Signal Processing Program contains an integrated set of routines which analyze linear and non-linear systems and circuits and their effects on user specified time domain waveforms. Based on a 512 point Fast Fourier Transform and its inverse. Linear processing is in frequency domain and non-linear processing is in time domain.
SPP (CP/M & MS-DOS) $69.95

SOFTWARE UTILITIES

Autodiff: File difference detector. This program finds insertions, deletions, and changes between any two files. Autodiff can mark the file, display, or print the differences, and more!
Autodiff (CP/M) $29.95

CP/M DateStamper: Automatically stamp your files with the date it is created, last read, or modified. Works without a Real Time Clock , or with many clocks currently on the market. Utilities are included to allow copying, erasing, or renaming files based on time and date. A time logging utility is included to record computer usage for business/tax purposes.
DateStamper (CP/M) $49.95

Media Master +: Read and Write up to 75 CP/M, MS-DOS & TRSDOS disk formats on your IBM or look-alike computer. ZP/EM program is bundled with Media Master to allow CP/M programs to run directly on your MS-DOS computer. An $80.00 value.
Media Master + ...$59.95

Pack and Crypt: Two program set. Pack compresses and expands files on disk to save space. Crypt encodes files to provide security for sensitive data. Both are ideal for use with modem transfers.
Pack and Crypt (CP/M & MS-DOS) $24.95

Sidekick: One of the most popular programs ever written. Use Sidekick as a calculator, notepad, appointment calendar, auto dialer, ASCII conversion table and much more. On-line help if you forget any of Sidekick's many functions.
Sidekick (MS-DOS) $54.95

SmartKey II: New Release! Same great time saver as the original, and allows compiling of definitions you set up with your word-processor! Makes every software program you use easier. Can reduce keystrokes by more than 50% by redefining any key on your keyboard to be any combination of characters or commands that you desire.
SmartKey II $49.95

SmartPrint: A powerful add-on to SmartKey, SmartPrint is a vesatile writing tool designed to give you full access to your printer's features such as wide, bold, condensed, underlined, subscript, superscript, and more. Works great with programs like WordStar and others.
SmartPrint $29.95

Uniform: Your Computer can read and write up to 80 CP/M, MS-DOS / PC-DOS & TRSDOS disk formats. Versions available for most popular CP/M and MS-DOS computers. Specify your host computer when ordering.
Uniform (CP/M & MS-DOS) $69.95

XTREE: Directory maintenance program that graphically displays subdirectories and filename paths. Complete control of your directory including delete, rename, view, list or show. A must for your IBM or compatible.
XTREE (MS-DOS) $49.95

Super Zap: Disk patch and dump program. If you have used DU, you will love this menu driven marvel!
Super Zap (CP/M) $24.95

ZP/EM: Run almost any CP/M program on your IBM or clone. Use with Media Master or Uniform to allow programs on CP/M disk formats to run directly on your IBM or compatible computer.
ZP/EM (MS-DOS) $39.95

FX, QX-10, PX-8 - Epson Corp; CP/M - DRI; MS-DOS - MicroSoft; PC-DOS - IBM Corp.; dBASE II & dBase III - Ashton-Tate; WordStar - MicroPro; UNIX - Bell Laboratories; Apple - Apple Computer Inc.; Basic Business - Advent Products Inc.

PROGRAMMING LANGUAGES

C/80 Ver 3.1: Full featured C compiler and runtime library. One of the fastest on the market. Mathpak is included for true 32 bit floating point and signed integers.
C/80 Ver. 3.1 (CP/M) $79.90

C/NIX: Operating System Enhancement for CP/M. C/NIX gives your system many features in the UNIX OS such a hierarchical directory, I/O redirection, "pipes" & "filters" and command files. Uses only 2.3K of TPA and 42K of disk. Requires CP/M 2.x.
C/NIX $59.95

LISP/80: Experiment with the artificial intelligence language. Based on the INTERLISP dialect, LISP/80 offers over 75 built-in functions, including file I/O, and string operations. Complete with 36 page manual and demo programs.
LISP/80 . . . . $39.95

Toolworks C: This compiler is a complete subset of C. The two- pass compiler produces relocatable object files (.obj) which are compatible with the MS-DOS LINK program. Mathpak is included for true 32 bit floating point and signed integers.
Toolworks C Compiler (MS-DOS) $79.90

Turbo Pascal: Borland version 3.0. The best Pascal compiler on the market.
Turbo Pascal (CP/M & MS-DOS) $69.95

Turbo Toolbox: Set of 3 utilities for use with Turbo Pascal.
Turbo Toolbox (CP/M & MS-DOS) $54.95

Turbo Tutor: Teaches step-by-step how to use Turbo Pascal.
Turbo Tutor (CP/M & MS-DOS) $34.95

Turbo Graphics: Provides full graphics management for producing windows, pie and pie charts, circles and other geometric shapes with Turbo Pascal.
Turbo Graphics (MS-DOS) $54.95

TEXT EDITING

Punctuation & Style: Improves your writing by catching unbalanced quotes, parentheses and brackets, improper abbreviations, capitalization, sentence structure, much more. It's like having your own copy editor!
Punctuation & Style (CP/M & MS-DOS) $125.00

Word Finder: This powerful 90,000 word Thesaurus allows you to select the best word for the application. Works inside WordStar for greater ease of use. Instantly searches its dictionary, then displays synonyms, and automatically deletes the "wrong" word and replaces it with the "right" word. Requires 380K disk storage.
Word Finder (CP/M) $79.95

Wordpatch: Print files with tiny, compressed, wide, or wide compressed type faces, 5 sizes of italic, real superscripts and subscripts, and 6, 7, and 8 lines per inch spacing. No new print controls to learn. Supports most popular dot matrix printers. A must for WordStar users!
Wordpatch (CP/M & MS-DOS) $49.95

The Word Plus: The ultimate spelling checker. Not only finds misspelled words but shows you correct spelling options, shows the word in context, allows you to build dictionaries of special words you use, and much more.
The Word Plus (MS-DOS) $150.00

HARDWARE & SUPPLIES

Finger Print "Letter Writer": Unleash your Epson FX series printer. Add near-letter-quality print, IBM and/or Apple Graphics printer emulation, plus 16 other print functions! Three replacement chips quickly fit inside Epson FX series printers. Easy installation. Does not void printer warranty.
Finger Print "LetterWriter" $79.95

Diskettes, Double Density:

Maxell 10-pack w/ storage box:		3M box of 10:	
Single Sided	**$19.95**	**Single Sided**	**$22.95**
Double Sided	**$23.95**	**Double Sided**	**$26.95**

Economy Diskettes: package of 25 including tyvek sleeves.
Single Sided $29.50
Double Sided $31.25

Call or write for our FREE catalog

All items are warranteed for 90 days. 30 day money back guarantee if not completely satisfied. Guarantee for software applies only if diskette seal is intact. Visa and MasterCard are welcome. Please add 2.00 freight per total order and 2.00 for COD orders. California residents please add 6% sales tax. Prices, availability and specifications subject to change without notice.

CALL TODAY **National (800) 821-8778**
California (800) 521-7182

Hours: Mon - Fri 8 am - 5 pm PDT

DEALER INQUIRIES WELCOME

advent products inc.

3154-F E. La Palma Ave
Anaheim, CA 92806
(714) 630-0446

get offsets list.

The name field addresses of all verbs in the host are entered into the nfa list by the verb NFALISTFILL. NFALISTFILL fills this list in the sequence followed by the recursive-branch-searching routine when searching the host dictionary (starting at the verb pointed to by FORTH CONTEXT @ @). This list is not necessarily in the same sequence as the physical order of the dictionary, and thus must be sorted. Since the list usually has large areas that are already in sequence with just the order of these blocks out-of-sequence, the sort can be performed rapidly by the verb WINDOWSORT. This routine detects the limits of the out-of-sequence blocks with WINDOWTOP, FRAME-BOTTOM, and WINDOWBOTTOM. The blocks are then moved into sequence by OPENMOVE.

The relocation routines compare the host Forth image generated by TARG1 (in the DOS file TARGET.TMP) with the current resident image (TARG2). These images should differ only for addresses, and the addresses should be different by exactly 201H. All addresses within the verbs being included in the target machine code must be recalculated to account for the squeezing out of unwanted verbs in the host. Certain addresses at the beginning of the target code must also be patched to reflect the memory map of the target machine.

Either of two pairs of routines direct this relocation operation, depending on whether headers are to be included in the target code (either RELOCATE and VERB>TARG, for headers, or RELOCATENH and VERB>TARGNH, for no headers). RELOCATE directs DISCARD to advance the file containing the image of TARG1 past areas of the file that are to be discarded. The verb VERB>TARG then copies the verb currently being examined by RELOCATE to the target image file. All addresses detected by VERB>TARG are examined by REPAIRADD, which returns the address reflecting the relocation of the current verb. If a link address is being adjusted, REPAIRADD directs NEWLINK to determine a new link address to account for intervening verbs in the host that are not to be included in the target code. In this process, the host dictionary structure is lost and the target dictionary links will reflect the physical order of its verbs. REPAIRADD detects whether the addresses need to be patched by PATCHADD to take into account the memory map of the target machine. REPAIRADD then consults the target offsets list to adjust the addresses to reflect the relocation of code.

If numbers in the two images are not equal, the verb VAR examines the code to determine if the number is in the data area of a variable. If this is not true, an error is issued by VAR and a dump produced by DUMP/=. In this case the tagged verbs are all untagged to allow examination of the host system from the outer interpreter by the application programmer.

If no headers were to be included, the same steps as above occur except that RELOCATENH and VERB>TARGNH are used. Because headers are not to be included in the target code, the vocabulary links do not need to be recalculated.

ARE YOU TRYING TO COMMUNICATE ?

C programs can communicate with the world now through the power of **The Greenleaf Comm Library.** Now from the people who brought you **The Greenleaf Functions** General Library for C, comes this rich interrupt driven, ring-buffered asynchronous communications capability.

Over 100 functions in C and assembler to facilitate communications at up to 9600 baud. Up to eight ports at a time. ASCII or XMODEM. X-On/X-Off too. Hayes compatible modems controlled here. Safe too, bet you can't exit your application with interrupts hot. Major applications around the world base their communicating applications on **The Greenleaf Comm Library.** Stop just trying and start really communicating. Get your copy of **The Greenleaf Comm Library** today. For all major C compilers, all models, all versions. For the IBM PC and just about any machine with MSDOS and an 8086. Comes with source code, extensive examples, demo programs, featuring **C-Terminal,** reference card and newsletter. No royalty. $185

Other Products: The Greenleaf Functions General Library, over 220 functions for total control of the IBM PC, with source. $185 for the compilers listed below. (See ordering instructions below).

Specify compiler when ordering: Lattice, Microsoft, Computer Innovations, Mark Williams, or DeSmet. Add $7.00 for UPS Second Day Air (or $5.00 for ground). Texas residents add sales tax. Mastercard, VISA, check, or P.O. In stock, shipped same day.

☐General Libraries	$185
☐Comm Library	$185
☐CI186 Compiler	$349
☐Lattice C	$395
☐Mark Williams	$475

PRICES ARE SUBJECT TO CHANGE WITHOUT NOTICE.

For Information:
214/446-8641

GREENLEAF SOFTWARE©

2101 HICKORY DR.
CARROLLTON, TX 75006

Circle no. 43 on reader service card.

After all tagged verbs have been relocated and the application code file closed, a map of the memory allocation of the target machine is displayed by TARGETMEMORY. During the relocation process, each verb to be included in the target code is displayed on the screen along with its address in the target machine. This information is also copied to a file if requested (1 ' MAPFILE !) by the application programmer.

The DOS file interface is identical for both the CP/M-80 and MSDOS versions. The primary verbs are FC@ and FC!. These verbs fetch or store a byte from the sequential DOS file specified by the file specification address on the stack. The defining verb FILE creates a verb that functions like a verb created by VARIABLE. The verb created by the defining FILE has three parts: first, a count index byte; second, a disk memory area (128 bytes); and third, a DOS filename specification area (file control block). When the verb is executed at run-time, the address of the count byte is left on the stack. The verb DMA will advance this address to the disk memory area, and the verb FCB will advance the count byte address to the DOS filename specification area (file control block). The verbs FILE@ and FILE! are used by FC@ and FC! to fetch or store the next sequential segment to or from the disk memory area. Note that the file areas can be reused as often as desired. The verbs SPACEFILE, SPACESFILE, CRFILE, TYPEFILE and U.RFILE perform output functions to a file that are the same as analogous screen display verbs.

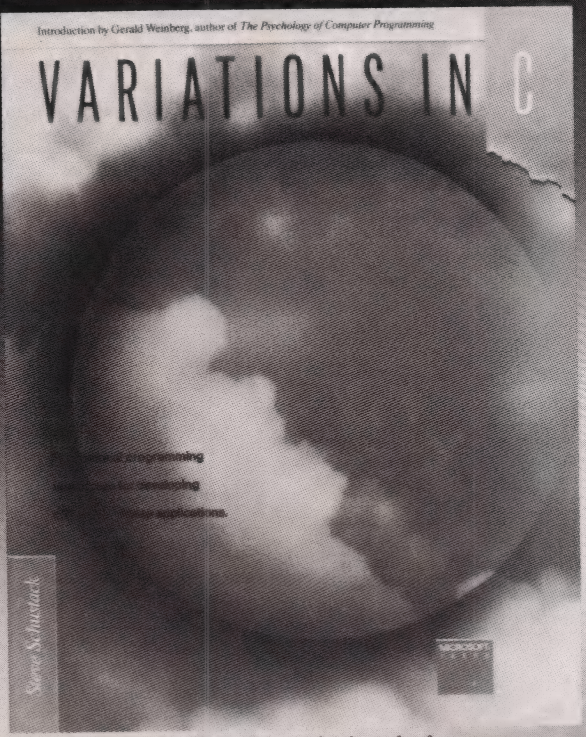
There are other verbs involved in maintaining the DOS file directory and in opening and closing DOS files. These verbs all return status flags except SETDMA. The verb FILENAME? queries a DOS filename from the keyboard and then sets up the filename specification area to be ready to search for or open a DOS file. File specification can also be loaded by the string-handling verb PUT$ with string verbs created by the defining verb STRING.

Complex Forth Constructions

Forth allows great flexibility in generating new programming constructions. Rather than trying to anticipate all possible new constructions, we decided to have the recompiler handle simple cases, but to allow the application programmer easy access to change the recompiler to handle more complex cases. Examples of cases that the recompiler does not currentiy handle are: 1.) a new verb with the function of colon; 2.) verbs that have an unusual effect on the parameter field of a verb being compiled (such as branches and literals).

The recompiler does properly handle simple CREATE-DOES> constructions (such as the defining verb STRING, screen #125). A verb such as F$ (screen #128) would be included in the application code if it had been referenced. The application code would also include the defining verb (STRING) since this verb contains the run-time action of the defined verb (F$). All other verbs necessary for the function of the DOES>

VARIATIONS IN C



$19.95 Available wherever books and software are sold or order directly from Microsoft Press.

An Important C Reference for Every Applications Programmer

If you develop business or professional applications in C, you'll want to have VARIATIONS IN C at your side. New from Microsoft Press, VARIATIONS IN C will help you extract all of C's incredible power and flexibility. Author Schustack's thorough and systematic approach to C's basic syntax and advanced programming techniques emphasize the importance of the user interface and ease of maintenance. His examples, including a functional order-entry application, are immediately useful as well as instructive.

VARIATIONS IN C is based on the proposed ANSI standards, using MS-DOS and the Microsoft C compiler, version 3.0.

ORDER TODAY

YES... please send me ________ copies of VARIATIONS IN C at $19.95 (plus $2 postage & handling) each.* 0-914845-48-9

*California residents add 6.5% sales tax; Washington state residents add 8.1% sales tax.

Name ____________________

Address ____________________

____

City ____________ State ______ ZIP ______

☐ Check ☐ VISA ☐ MasterCard ☐ American Express

Credit Card No. ____________ Exp. Date ______

Send coupon along with payment or credit card information to:
Microsoft Press
10700 Northup Way, Box 97200
Bellevue, WA 98009

DDA5

MICROSOFT PRESS

Circle **no. 72** on reader service card.

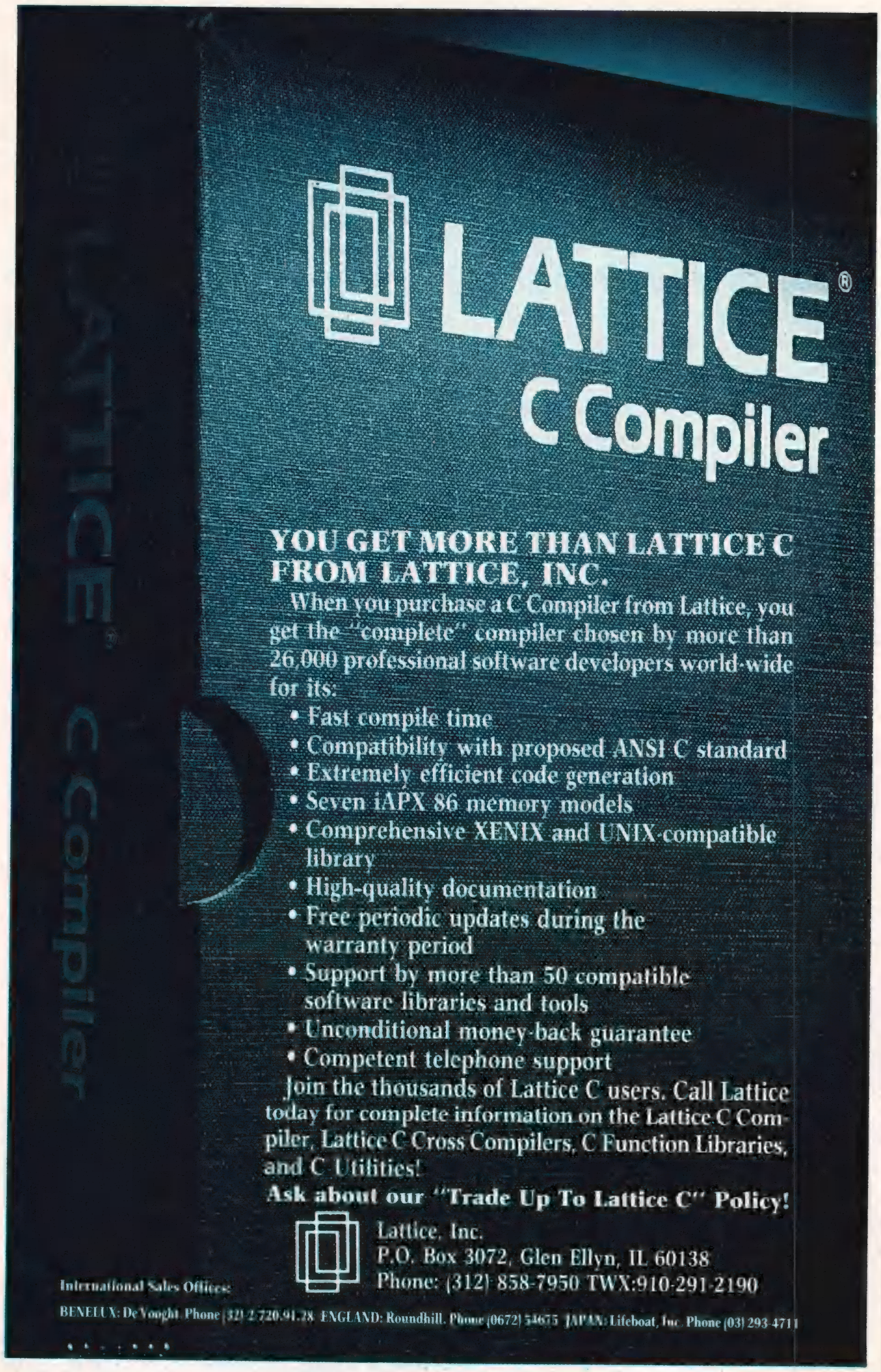
portion of the defining verb would also be included.

If the recompiler finds a value in a code field of a verb that it does not recognize, then that verb will be included in the application code; however, no further searching of this verb occurs. If this verb was the result of a new colon-type definition, then not searching this verb may result in some verbs that are needed not being included in the application code.

The recompiler may not generate the proper application code if there are bytes in the parameter field of a colon-defined verb that are not a code field address (or special instances that the recompiler has been instructed to search for, for example, branches and literals).

The recompiler has been designed to allow cases such as those described above to be easily included in the recompiler by the application programmer. The verb PFSKIPS on screen #137 must be augmented to handle special bytes within the parameter field of colon-defined verbs. The verb TREETAG on screen #139 can be augmented to handle cases such as newly defined colon-type definitions.

The verb EXIT must not be used since the verb tagging routines assume that the cfa of the verb EXIT terminates each colon defined verb. The cfa of EXIT must be the last cfa in the parameter field of a colon verb and must not appear anywhere else in the definition.



YOU GET MORE THAN LATTICE C FROM LATTICE, INC.

When you purchase a C Compiler from Lattice, you get the "complete" compiler chosen by more than 26,000 professional software developers world-wide for its:

- Fast compile time
- Compatibility with proposed ANSI C standard
- Extremely efficient code generation
- Seven iAPX 86 memory models
- Comprehensive XENIX and UNIX-compatible library
- High-quality documentation
- Free periodic updates during the warranty period
- Support by more than 50 compatible software libraries and tools
- Unconditional money-back guarantee
- Competent telephone support

Join the thousands of Lattice C users. Call Lattice today for complete information on the Lattice C Compiler, Lattice C Cross Compilers, C Function Libraries, and C Utilities!

Ask about our "Trade Up To Lattice C" Policy!

Lattice, Inc.
P.O. Box 3072, Glen Ellyn, IL 60138
Phone: (312) 858-7950 TWX:910-291-2190

International Sales Offices:
BENELUX: De Vooght. Phone (32) 2-720.91.28 ENGLAND: Roundhill. Phone (0672) 54675 JAPAN: Lifeboat, Inc. Phone (03) 293-4711

Circle **no. 36** on reader service card.

Availability of the Ready-to-Run Target Compiler from CICFIG

This project grew out of the needs of several of the CICFIG members for a target compiler to use to generate PROMable code for our work on various instrumentation control applications. We are placing the result of our efforts in the public domain because we believe there is a broad-based need for such a package. This target compiler fulfills a large part of our initial desires. It has not been ported to as large a variety of microprocessors as we had initially hoped, but this may change after the release of this code.

The address relocation process is not as fast as we had hoped; however, the modular design of the recompiler will allow us easily to replace these routines with more efficient ones in a later release.

Since the generation of TARG1.COM and TARG2.COM may be a bit complicated due to the recoding of certain parts of the kernel of the host Forth system, we have decided to distribute (at a cost of $30) a package containing the target compiler and all the parts of the target compiler generator. This package contains: the ready-to-run target compiler (TARG1 and TARG2), the verb tag tracer, and a program to filter the CP/M-80 target code to move the return stack pointer to RAM. Also included are the sources (Forth blocks and DOS files) for the recompiler, the examples, and the kernels. Documentation on the generation of the target compiler and its use to target-compile application code are also included.

At present we are supplying this material for use on MSDOS compatible computers and on CP/M-80 compatible systems. Other systems may become available later and we should be contacted. Disk formats which are available are MSDOS Version 2 IBM-PC DSDD 5¼-inch 8-sector, CP/M-80 Version 2.2 standard 8-inch and CP/M-80 NorthStar Horizon SSDD 5¼-inch.

Address of CICFIG

CICFIG
Dept. of Electrical and Computer Engineering
University of Illinois
1406 W. Green St.
Urbana, Illinois 61801

Acknowledgements

We thank the members of the Target Compiler Group and the other members of CICFIG for their input into the design and testing of this target compiler. Everett Dade provided invaluable assistance by rigorously examining the source code. This work was supported in part by: a grant from the U.S. Department of Energy (DOE DEAC02 80ER10701) to Dr. Antony Crofts, a grant from the National Foundation for Cancer Research to Dr. Harold Swartz, and by a grant from the National Aeronautics and Space Administration under grant NsG–7506.

Glossary for the CICFIG Recompiler Source Code

The following abbreviations are used in the stack pictures.

a	address (16 bit addressing)
b	byte
cfa	code field address
f	flag (0 = false)
fa	file address (the address of the first byte of a data area created by the defining word FILE)
n	number (16 bit signed)
nfa	name field address
pfa	parameter field address

!FIX (--- a) scr# 122 A variable used to avoid patching the link field of the verb ! when headers are to be included in the target code.

0BRANCHCFA (--- cfa) scr# 124 A constant that leaves the cfa of 0BRANCH.

;S (---) scr# 126 Put a zero (null) at the next byte of the input stream to terminate interpretation of a screen.

<+LOOP>CFA (--- cfa) scr# 124 A constant that leaves the cfa of <+LOOP>.

<.">CFA (--- cfa) scr# 124 A constant that leaves the cfa of <.">.

</LOOP>CFA (--- cfa) scr# 124 A constant that leaves the cfa of </LOOP>.

<;CODE>CFA (--- cfa) scr# 124 A constant that leaves the cfa of <;CODE>.

<ABORT">CFA (--- cfa) scr# 124 A constant that leaves the cfa of <ABORT">.

<LOOP>CFA (--- cfa) scr# 124 A constant that leaves the cfa of <LOOP>.

>TTAG (---) scr# 136 Execute the verb TREETAG. This allows a forward reference to TREETAG during compilation of the recompiler.

APPTAG (---) scr# 140 This verb routes the verb tagging routines.

ASCII (--- b) scr# 127 Leave the number that is the ASCII code for the next character after space in the input stream.

BRANCHCFA (--- cfa) scr# 124 A constant that leaves the cfa of BRANCH.

CASE (---) scr# 127 Defined in "Eaker Case Statement" *FORTH Dimensions* II/3 p 37, 9/80.

CLEANUPDOS (---) scr# 155 Flushes and closes the DOS files used by the relocation routines.

COUNTVERBS (nfa ---) scr# 134 Search the dictionary of the host Forth system from the verb with name field address nfa to the beginning of the dictionary. Keep a running count of the number of verbs encountered in the variable VERBS#, and the number of tagged verbs in the variable TVERBS#. This verb is recursive and is started by leaving the nfa obtained from FORTH CONTEXT @ @. The entire host Forth system is searched through recursive calls to COUNTVERBS to include all vocabularies.

CREATE-DOES> (a ---) scr# 138

THE BEST Z80 ASSEMBLER ON THE MARKET JUST GOT BETTER!

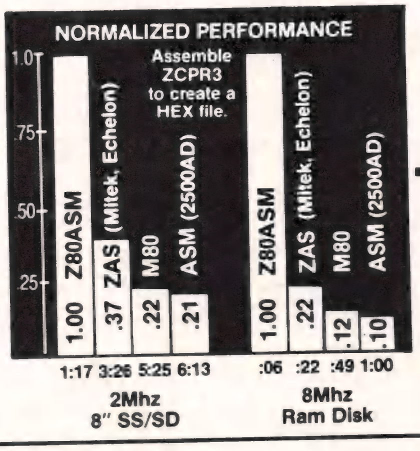
Z80ASM

NOW ONLY **$49.95**

DON'T ASK HOW OURS CAN BE SO FAST . . . ASK WHY THEIRS ARE SO SLOW!

". . a breath of fresh air . ."
Computer Language, Feb. 85

". . in two words, I'd say speed & flexibility",
Edward Joyce, User's Guide #15



Now fully compatible with M80 in .Z80 mode with many extensions. Time & date in listing, 16 char. externals, plus many other features.

To order, or to find out more about our complete family of development tools, call or write:

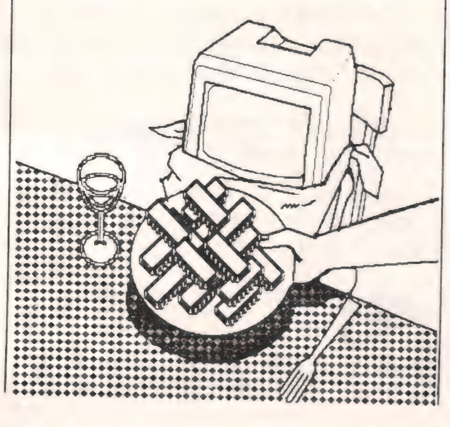
SLR Systems

1622 N. Main St., Butler, PA 16001
(800) 833-3061, (412) 282-0864
Telex 559215 SLR SYS

C.O.D., Check or Money Order Accepted

SHIPPING: USA/CANADA + $3 ▪ OTHER AREAS + $10
Z80 CP/M compatibility required.

Circle **no. 78** on reader service card.



Tag the verb with the largest nfa less than the address a. If a points to the DOES> portion of a verb that is a CREATE-DOES> construction, continue to TREETAG all verbs which the DOES> portion of this verb requires. Otherwise, indicate an exception ("--- Code verb assumed") and continue without searching the verb that the address a points into.

CRFILE (fa ---) scr# 134 Append a <cr> to the DOS file specified by the file address fa.

DDRIVE (---) scr# 141 Prompt the operator, and pause until a key is pressed.

DISCARD (n ---) scr# 152 Discard n bytes from the file IMG (TARGET.TMP).

DJMPLEN (--- n) scr# 120 An implementation specific constant that leaves the number of bytes of the machine code sequence that is a jump to DODOES at the start of the DOES> portion of a "CREATE DOES>" defining verb.

DMA (fa --- a) scr# 128 Leave the address of the disk transfer area of the file specified by the file address fa.

DOCODE (--- a) scr# 122 A variable used in TREETAG to determine if the verb being currently examined is a code level verb.

DOCOL (--- a) scr# 124 A constant that leaves the address of DOCOL.

DOCON (--- a) scr# 124 A constant that leaves the address of DOCON.

DOESJMP (--- b) scr# 120 An implementation specific constant that leaves the first byte of the machine code sequence that is a jump to DODOES at the start of the DOES> portion of a "CREATE DOES>" defining verb.

DOUSE (--- a) scr# 124 A constant that leaves the address of DOUSE.

DOVAR (--- a) scr# 124 A constant that leaves the address of DOVAR.

DOVAR? (--- a) scr# 122 A variable used as a flag by the code relocation routines to signal that a variable is currently being relocated.

DOVOC (--- a) scr# 124 A constant that leaves the address of DOVOC.

DOWN (--- a) scr# 122 A variable whose value is used as a pointer to within the name field address list.

DUMP (a n ---) (defined in MVP-FORTH Utilities) Display the contents of the n bytes of memory starting at address a.

DUMP/= (a ---) scr# 148 Display the contents of memory in the vicinity of the address a. This will help to identify the section of the Forth system that was not equivalent to the image in the DOS file TARGET.TMP. In addition, the DOS files are closed, all tagged verbs are untagged, and then the program aborts.

ENDCASE (---) scr# 127 See CASE.

ENDOF (---) scr# 127 See CASE.

EXITCFA (--- cfa) scr# 124 A constant that leaves the cfa of the verb EXIT.

**F$** (--- a n) scr# 128 A string created by the defining verb STRING. There are n bytes currently in F$, starting at address a. This string holds the name of the DOS file TARGET.TMP.

FC! (b fa ---) scr# 133 Append the byte b to the file specified by the file address fa.

FC@ (fa --- b) scr# 133 Get the next byte b from the file specified by the file address fa.

FCB (fa --- a) scr# 128 Leave on the stack the address of the file control block area specified by the file address fa.

FCLOSE (fa --- f) scr# 131 Close the DOS file specified by the file address fa. Leave the flag FFFF hex if the file cannot be found, 0 if successful.

FCREATE (fa --- f) scr# 130 Make (and open) a new DOS file with the name specified by the contents of the FCB area of the file specified by the file address fa. Leave FFFF hex if no more directory space is available, 0 if successful.

FDELETE (fa --- f) scr# 130 Delete the DOS file with the name specified by the contents of the FCB area of the file specified by the file address fa. Leave FFFF hex if the file was not found, 0 is successful.

FILE (---) scr# 128 A defining word that creates a verb that functions like a variable with 162 bytes. Use like VARIABLE: FILE <filename>. In the parameter field of <filename>, the first byte is used to keep a count on the disk memory transfer area from DOS (the 128 bytes where data transfers from DOS are stored), and the last 33 bytes are used as the DOS file control block for the DOS filename.

FILE! (fa --- f) scr# 132 Append the 128 bytes in the file transfer area of the file specified by the file address fa. Leave FFFF hex if the disk was full, 0 if the write was successful.

FILE@ (fa --- f) scr# 132 Read the next 128 bytes into the DMA area from the DOS file named in the FCB area of the file specified by the file address fa. Leave FFFF hex if no more data can be read, 0 if the read was successful.

FILECREATE (fa --- f) scr# 132 Make (and open) a new DOS file deleting if necessary an existing file with the same name. The DOS file is named by the contents of the FCB area of the file specified by the file address fa. Leave FFFF hex if no directory space was available, 0 if successful.

FILEEOF (fa --- f) scr# 132 Leave FFFF hex if the last allocated byte of the DOS file specified by the file address fa has been read; 0 otherwise.

FILENAME? (fa ---) scr# 129 Prompt the keyboard for a DOS filename. The filename is loaded into the FCB area of the file specified by the file address fa.

FINDTAG (a1 --- a2) scr# 146 Search the name field address list starting at address a1 and leave the the address a2 of the next location in the list that has a value that points to a tagged verb.

FIXLINK? (--- a) scr# 122 A variable used by the relocation routines when the target code is to contain headers. A non-zero value indicates that the next encountered address is a link and must be patched to correct for the possible absence in the target code of the verb to which the link used to point to.

FIXSMUDGE (---) scr# 147 This verb sets to zero the smudge bit of all verbs which have been tagged by the tagging routines.

FLIP (n1 --- n2) scr# 126 Swap the high and low bytes of the word on the stack.

FNSEARCH (fa ---) scr# 130 Have DOS search for the next file that matches the file named (wildcards are usually used) in the FCB area specified by the file address fa. The matching DOS filename is returned in the disk memory transfer area last assigned by the verb SETDMA. Leaves FFFF hex if no match is found, 0 if a match is found.

FOPEN (fa --- f) scr# 131 Open a DOS file with the name contained in the FCB area of the file specified by the file address fa. Leaves FFFF hex if the file cannot be found, 0 if the file was successfully found.

FORTHORG (--- a) scr# 120 An implementation specific constant that leaves the address of the origin of the current host Forth system.

FRAMEBOTTOM (a1 n --- a2) scr# 144 Search down the name field address list from the pointer a1 until the value in the list is less than the number n. Leave the pointer a2 for this entry.

FRAMEIN (--- n1...nj) scr# 142 Construct the framework for the verb lists on the stack by leaving zeros for the j cells required. Certain locations within the lists and certain pointers to the lists are initialized. The verb RAZE will return the stack to its depth prior to FRAMEIN. The number j is calculated from the variables VERBS# and TVERBS#.

FREAD (fa --- f) scr# 131 Read the next 128 bytes from the file specified by the file address fa. These bytes are transferred to the disk transfer area specified by the last call to the operating system (SETDMA). Leave FFFF hex if there is no more data available, 0 if the read was successful.

FSEARCH (fa --- f) scr# 130 Have DOS search for the file named in the FCB area of the file specified by the file address fa. Leave FFFF hex if no match is found, 0 otherwise.

FWRITE (fa --- f) scr# 131 Append 128 bytes from the last disk transfer area set (SETDMA), to the file named in the FCB area specified the file address fa. Leave FFFF hex if the disk was full, 0 if the write was successful.

HEADERS (--- n) scr# 123 A constant that leaves either 0 if the recom-

U S Software®

Software Development Products for Design Engineers

8085 Embedded 6809
Application Support
68000 Software Z-80
8051 8086
8096

FPAC/DPAC™
Floating Point Libraries
- IEEE Single & Double Precision Format
- Source Assembly
- Complete Math Functions
- Data Conversion Procedures

USX™
Multi-Tasking Executives
- Real-Time and ROMable
- Dynamic Task Management
- Priority Task Scheduling
- Memory Management

United States Software Corporation
5470 N.W. Innisbrook Place • Portland, Oregon 97229 • (503) 645-5043

Circle **no. 150** on reader service card.

piler is to generate headerless code, or the pfa of the root verb of the application program if headers are to be included in the target code.

HITKEY (---) scr# 141 Pause until a key is depressed on the console.

ID.FILE (nfa fa ---) scr# 135 Copy a verb name to the file specified by the file address fa.

IMAGESAVE (---) scr# 141 Copy an image of the current Forth system from FORTHORG to HERE to the DOS file TARGET.TMP.

IMG (--- fa) scr# 128 A file specifier created by the defining verb FILE. Leaves the address fa of the disk transfer area count byte.

ISFF (f1 --- f2) scr# 130 Leaves FFFF hex if f1 is FF hex, 0 otherwise.

ISZERO (f1 --- f2) scr# 130 Leaves 0 if f1 is 0, FFFF hex otherwise.

LB (n --- b) scr# 128 Leave the low byte of the number n.

LITCFA (--- cfa) scr# 124 Leave the cfa of the verb LIT.

LOADFCB (a1 n a2 ---) scr# 141 Zero 33 bytes starting at address a2 (an FCB area), and move n bytes from address a1 to address a2+1.

LTIB (--- a) scr# 120 A constant that leaves the address of a cell in low memory that contains a pointer to the terminal-input-buffer.

**M$** (--- a n) scr# 128 A string created by the defining verb STRING. There are n bytes currently in M$, starting at address a. This string holds the DOS filename attribute MAP.

MAP (--- fa) scr# 128 A file specifier created by the defining verb FILE. Leaves the address of the disk transfer area count byte.

MAPFILE (--- f) scr# 123 A constant that leaves a zero if no map file is to be generated; otherwise, signal the generation of a DOS ASCII file <filename>.MAP with the names and addresses of the verbs included in the target code.

MAXNFA (nfa1 a nfa2 --- nfa3 a) scr# 135 This verb recursively searches the entire host dictionary starting at nfa2 (FORTH CONTEXT @ @) and returns nfa3 which is the maximum nfa that is less than or equal to a. When started, nfa3 should be 0.

NEWLINK (nfa1 --- nfa2) scr# 148 Find the host name field address nfa2 of the verb that when moved to the target code, should be pointed to by the link field of the verb with the host name field address nfa1.

NEXTNFA (nfa1 --- nfa2) scr# 150 Leave the name field address nfa2 of the next verb in physical order in the host Forth system after the verb with the name field address nfa1. If the verb with name field address nfa1 is the last verb in the dictionary (in physical order), then leave HERE.

NFALISTFILL (nfa ---) scr# 143 Search the dictionary of the host Forth system from the verb with name field address nfa, to the beginning of the dictionary. Enter the name field address of all verbs encountered in the name field address list. This verb is recursive and is started by leaving the nfa obtained from FORTH CONTEXT @ @. The entire host Forth system is searched through recursive calls to NFALISTFILL to include all vocabularies.

NFTAG (nfa ---) scr# 136 Set the smudge bit of the verb with name field address nfa to 1.

NLIST (--- a) scr# 123 A constant that leaves the address of the first entry in the name field address list.

OF (---) scr# 127 See CASE.

OFFSETS (---) scr# 146 Enter into the tagged verb offsets list the name field addresses (or code field addresses) of all tagged verbs in the host Forth system. With each entry, store the difference between this address in host Forth system and the address that verb will be at in the target code.

OPENMOVE (a1 a2 a3 ---) scr# 145 Insert at list pointer a2, the section of the name field address list that starts at pointer a1 and ends at pointer a3. See WINDOWSORT.

PATCHADD (a1 --- a2 f) scr# 149 If the flag f is 1, then the address a2 found at the address a1 should be corrected to a new address for the target code. If the flag f is 0, then the address a2 should not be corrected. This is used to allow loading of target code RAM addresses.

PATCHMAX (--- a) scr# 120 An implementation specific constant

PCTEX™ **is here!**

Complete TEX82 Typesetting for your PC/XT or AT

- **Real, state-of-the-art typesetting capable of handling all mathematical and scientific material.**
- **Produce work of this quality on your Epson printer:**

$$\sum_{p\ \text{prime}} f(p) = \int_{t>1} f(t)\, d\pi(t). \qquad \underbrace{\{\overbrace{a,\ldots,a}^{k\ a\text{'s}},\overbrace{b,\ldots,b}^{l\ b\text{'s}}\}}_{k+1\ \text{elements}}.$$

- **Produce this quality work on a Corona Laser Printer:**

$$G(z) = e^{\ln G(z)} = \exp\left(\sum_{k\geq1} \frac{S_k z^k}{k}\right) = \prod_{k\geq1} e^{S_k z^k/k}$$

- **PCTEX includes a 120-page beginners guide and macros, and the LaTEX Document Preparation System and macros.**
- **PCTEX is a full implementation of Donald Knuth's TEX82.**
- **PCTEX: only $279. PCDOT dot-matrix printer driver: $100. (Printers: IBM Graphics, Epson FX, RX, LQ1500, Toshiba.) PCLaser printer drivers: $300. (For Corona, QMS, Apple.) Drivers include over 200 fonts.**

Requires DOS 2.0 or better, 512K RAM, 10M hard disk.

PERSONAL TEX INC

20 Sunnyside, Suite H, Mill Valley, CA 94941.
(415) 388-8853. Telex 275611.

TEX: American Mathematical Society. PCTEX: Personal TEX, Inc.. IBM-PC: IBM Corp.. QMS: QMS, Inc

Circle no. 76 on reader service card.

that leaves the address above which in the host Forth system no addresses will need to be patched by PATCH-ADD for inclusion in the target code.

PFSKIPS (a1 --- a1 or a2 0) scr# 137 Check to see if the cfa (which is an entry in the parameter field of a colon verb) pointed to by a1 is in need of special handling, such as branches, loops and text. Leave the address a1 if this cfa is not an exception. Leave the address a2 and a 0 flag to bypass searching the verb referenced by a1, and continue searching at the pointer a2 + 2.

PUT$ (a n ---) scr# 125 Move the bytes (string) in the input stream starting after the next space character to the next $ character (exclusive) to the address a. The byte before the address a stores the current number of bytes in the string and the byte below that holds the maximum number of bytes in the string (see STRING). QUIT if the string is too long. The number n is ignored.

RAZE (n1...nj ---) scr# 142 Remove the verb lists (j cells) from the parameter stack (see FRAMEIN). The number j is calculated from the variables VERBS# and TVERBS#.

RELOCATE (--- n) scr# 152 Used when headers are to be included in the target code. This word organizes the relocation of the necessary sections of the host Forth system to generate the target code. The total number of bytes n in the target code is left on the stack.

RELOCATENH (--- n) scr# 154 Used when headers are not to be included in the target code. This word organizes the relocation of the necessary sections of the host Forth system to generate the target code. The total number of bytes n in the target code is left on the stack.

REPAIRADD (a1 --- a2) scr# 150 Leave the address a2 which is the value at the host Forth address a1, corrected to reflect the address relocation of the code to be included in the target code.

ROOF (--- a) scr# 123 A constant that leaves the address of last name field address in the name field address list.

SETDMA (fa ---) scr# 131 Instruct DOS that the disk transfer area is at memory location fa+1.

SETUPDOS (---) scr# 155 Setup the DOS files for the relocation routines of the recompiler.

SFLAG (f1 n --- f2 n) scr# 136 This verb changes the flag f1 to f2 (f2=f1-1), This is used to determine the none-of-the-above case of the CASE construction in the verb TREETAG.

SPACEFILE (fa ---) scr# 134 Append a space character to the file specified by the file address fa.

SPACESFILE (n fa ---) scr# 134 Append n space characters to the file specified by the file address fa.

STRING (n ---) scr# 125 A defining word used to create a dictionary entry for the verb with the name of the next character string in the input stream. The new verb will be allotted n+2 bytes. The first byte is the maximum allotted space for the string, the next byte is the current length of the string. At run-time the new verb will leave on the stack the address of the first byte of the string and the number of bytes in the string.

T'STREAM (--- a) scr# 129 Functions as 'STREAM except that the verb BLOCK has been removed

T#BUFF (--- n) scr# 123 A constant that leaves the number of disk block buffers required by the application program.

TARG (--- fa) scr# 128 A file specifier created by the defining verb FILE. Leaves the address of the disk transfer area count byte.

TARGET (pfa or 0 ---) scr# 157 This verb routes the operation of the target compiler. If the top of the stack has a zero, then no headers will be included in the target code; otherwise, this value is the parameter field address of the root verb of the application program and headers will also be included in the target code.

TARGETMEMORY (n ---) scr# 156 Display a memory map of the target code that has n bytes.

TARGMAP (n nfa ---) scr# 148 Display the verb name with name field address nfa and display the target code address (n + TARGORG) of that verb. Copy this same information to a DOS file if a map file has

9 TRACK TAPE CONTROLLERS AND ½" TAPE SUBSYSTEMS

MODEL TC-PC

TC-PC is a high performance 9-track tape controller for the IBM-PC with these important features:

- Reads and writes industry standard ½-inch tape
- Compatible with most formatted tape drives
- Standard 8-bit parallel recording with parity, and read-after-write verification
- Switch selectable I/O address (four contiguous ports required for operation)
- Maximum data transfer rate of 192,000 bytes per second
- Record length from 1 to 65,535 bytes
- Supports up to 8 tape transports
- Jumper selectable DMA channel
- Modes: PE and NRZI at 800, 1600, 3200 and 6250 bytes/inch
- Installable device drivers allow creation of application programs which run under IBM XENIX and MS-DOS
- Operates with IBM-PC and -XT; Compaq Portable; Zenith PC-150; Sperry PC; the Leading Edge Computer, and other 100% IBM-PC compatible equipment.

MODEL TC-50

TC-50 offers all the standard features of the TC-PC with these additional enhancements:

- Maximum data transfer rate of 400,000 bytes/second; 904,000 bytes/second with memory option
- Operation with a wider range of IBM-compatible machines, including IBM-AT; Compaq Desk Pro; ATT 6300 and others

XENIX and MS-DOS are Registered Trademarks of Microsoft Corp.
IBM-PC/AT/XT are Registered Trademarks of International Business Machines Corp.
Compaq Portable and Compaq Deskpro are Registered Trademarks of Compaq Corp.
AT&T 6300 © AT&T Information Systems Corp.
Sperry PC © Sperry/Univac Corp
Zenith PC-150 © Zenith Data Systems
Leading Edge is a Registered Trademark of Leading Edge Products, Inc.

A variety of software utilities is supplied as part of the TC-PC and TC-50 packages, including:

- **DEPOT (Data Exchange Program with Optional Translation)**
  DEPOT provides a means to transfer data between system disk and magnetic tape, allowing:
  - Data interchange from tape to disk, and disk to tape
  - Conversion from ASCII to EBCDIC, and vice versa
  - Positioning to arbitrary location prior to data read
  - Specification of record length and block factor when writing from disk to tape; allows deblocking when reading from tape to disk
  - Multiple operations to be specified from a command file
- **TAU (Tape Archive Utility)**
  - Provides individual file backup and restore
  - Allows use of MS-DOS wild cards such as "*.*"
  - Provides disk drive selections for I/O
  - Changes pathname selections from within TAU
  - Provides data encryption for security

WARRANTY
All Overland Data products carry a 30-day unconditional money-back guarantee, and are warranted for one year, parts and labor.

OVERLAND DATA, INC.
5644 Kearny Mesa Road #A
San Diego, CA 92111
Tel. (619) 571-5555

Circle **no. 39** on reader service card.

been requested.

TARGORG (--- a) scr# 123 A constant that leaves the address of the origin of the target code.

TBUF1 (--- a) scr# 123 A constant that leaves the address of the start of the disk block buffer area of the target code.

TCOLD1 (--- a) scr# 120 An implementation specific constant that leaves the address of the cell in low memory that contains the address of the first verb to be executed upon start-up of the target code.

TCOLDCFA (--- cfa) scr# 123 A constant that leaves the cfa of the first verb to be executed upon start-up of the target code.

TEM (--- a) scr# 123 A constant that leaves the address of the top of the disk block buffer area in the target code.

TFIX;CODE (---) scr# 140 This verb tags certain ;CODE verbs that are mandatory in the target code.

TFIXNULL (---) scr# 140 This verb tags the verb X (null) and then patches the header for this verb.

TLIST (--- a) scr# 123 A constant that leaves the address of the start of the target verb offset list.

TPICK (n1 --- n2) scr# 126 Functions as PICK except that ABORT" has been removed.

TR0 (--- a) scr# 123 A constant that leaves the address of the start of the target code return stack pointer for initialization of the inner-interpreter of the target machine.

TROLL (n ---) scr# 126 Functions as ROLL except that ABORT" has been removed.

TREETAG (cfa ---) scr# 139 This recursive verb tags (sets the smudge bit to 1) all verbs in the host Forth system that are required for the function of the verb whose code field address is cfa.

TREETAGCFA (--- cfa) scr# 123 A constant that leaves the cfa of TREETAG. Used to allow forward referencing at compilation time within TARGET.

TSP0 (--- a) scr# 123 A constant that leaves the address of the start of the target code parameter stack pointer for initialization of the inner-interpreter of the target machine.

TVARIABLE (---) scr# 126 A defining word used in the form: TVARIABLE <varname>. When TVARIABLE is executed, it creates the verb <varname> in the host dictionary with its parameter field initialized to the next allotted cell in the variable RAM area of the target machine. This allows variables to be implemented in RAM when the target code is in ROM. The application programmer must set up the target variable RAM area by loading the constant TVRAM.

TVERBS# (--- a) scr# 122 A variable used to hold the number of verbs that have been tagged by the tagging routines.

TVRALLOT (n ---) scr# 126 Add the number n to the variable TVRAMOFF. This allows target variable RAM bytes to be allotted.

TVRAM (--- a) scr# 123 A constant that leaves the address of the start of the target variable RAM area. This value must be loaded by the application programmer if needed.

TVRAMOFF (--- a) scr# 122 A variable that contains the current number of bytes of the target variable RAM area that have been allotted.

TWORD (b --- a) scr# 129 Functions as <WORD> except that ABORT" has been removed.

TYPEFILE (a n fa ---) scr# 134 Copy n characters starting at address a to the file specified by the file address fa.

U.RFILE (n1 n2 fa ---) scr# 134 Convert the number n1 to an unsigned ASCII number right justified n2 spaces. Append these ASCII characters to the file specified by the file address fa.

U8 (--- a) scr# 120 A constant that leaves the address of the cell in low memory that contains the value to initialize the eight user variable.

VAR (a ---) scr# 149 If the address a is the pfa of a VARIABLE defined verb (or the return stack pointer cell address) then copy the contents of that cell to the DOS file specified by TARG. Otherwise, dump the contents of the host Forth system in vicinity of the address a and indicate an error.

VERB>TARG (nfa ---) scr# 151 The host Forth verb (including the header) with name field address nfa is copied to the DOS file specified by TARG. Address relocation and patches are performed.

VERB>TARGNH (cfa ---) scr# 153 The host Forth verb (excluding the header) with code field address cfa is copied to the DOS file specified by TARG. Address relocation and patches are performed.

VERBLISTS (---) scr# 147 This routine routes the construction and filling in of the verb lists on the parameter stack.

VERBS# (--- a) scr# 122 A variable used to hold the number of verbs that are in the host Forth system dictionary.

VERBTAG (a ---) scr# 136 Tag (set the smudge bit to 1) the verb whose code field address or DOES> jump address is a.

WINDOWBOTTOM (a1 a2 --- a3) scr# 144 Examine the values in the name field address list starting with the element pointed to by a1. Continue so long as they are decreasing and greater than the contents of the element of the list pointed to by a2. When this is not true, leave a3, which points to the element just beyond this point.

WINDOWSORT (a ---) scr# 145 Sort the list of unsigned numbers (name field addresses) between the high memory address a and low memory location specified by the value FFFF hex in the list. At the end of the sort, change the entry at the bottom of the list to FORTHORG.

WINDOWTOP (a1 --- a2) scr# 144 Index down the list of unsigned numbers starting at pointer a1 until the values in the list are no longer monotonically decreasing. Leave the pointer a2 to where this occurs. See WINDOWSORT.

X (---) scr# 140 This is a pseudonym for null that does not have the compile-time parts of the host system Forth verb null. This version of null will be present in the target code.

DDJ

(Listing begins on page 76)

Reader Ballot
Vote for your favorite feature/article.
Circle Reader Service **No. 193.**

Who Says You Can't Tell A Book By Its Cover?

Dr. Dobb's Sourcebook: A Reference Guide for the C Programming Language

For years, serious programmers have relied on Dr. Dobb's Journal for the technical tools of their trade. Now, Dr. Dobb's presents the definitive programmers guide to the who, what, where, when and why of C, the leading language among software developers. This comprehensive guide to new information, products and services specific to C will be most often-used reference!

In this valuable guide you'll find:

- An extensive directory of hardware and software services—including classes and seminars, C programming opportunities, and on-line services
- A bibliography with over 300 listings of available articles and books on C
- A comprehensive C product listing—including C compilers, graphics modules, utilities, editors and development systems, and more!
- And much more practical C programming information

At only $7.95, no C programmer can afford to be without this unique reference.

TO ORDER: Mail this coupon, along with payment, to: **Dr. Dobb's Journal, 2464 Embarcadero Way, Palo Alto, CA 94303**

PAYMENT MUST ACCOMPANY YOUR ORDER

____ I enclose check/money order

____ Please charge my __VISA __M/C __American Express

Card #________________ Exp. Date ____

Signature ________________

Name ________________

Address ________________

City ____________ State ____ Zip ____

Please allow 6 to 12 weeks for delivery

Please send me ______ copies of **Dr. Dobb's Sourcebook**

at $7.95 each = ________

+ Shipping & Handling = ________

(Must be included with order. Please add $1.50 per book in U.S. $3.25 each surface mail outside U.S. Foreign airmail rates available on request.)

TOTAL = ________

3108

Procedure

TARG1 (from TARG1.COM DOS)

(A) load screen with memory map (scr# 200, example #1)
(B) load screen(s) with application (scr# 201, example #1)
(C) load screen with cold start (scr# 201, example #1)

0 TARGET (0 => no headers)

(control will be returned to the operating system)

TARG2 (from TARG2. COM DOS)

(A-C) exactly as above (200 201 THRU)

0 TARGET (0 => no headers)

(upon request, supply the DOS filename for the target code)
(control will be returned to the interpreter when complete)

Forth Compiler Listing **(Text begins on page 52)**

```
SCR #200
( No headers, No block verbs; Target Memory Map Constants) HEX
( This program outputs to printer, ASCII chars typed in in hex)

1 ' MAPFILE !

6000 ' TVRAM !
6000 ' TEM !
TEM ' TBUF1 !
TBUF1 52 - ' TRO !
TRO A0 - ' TSPO !
100 ' TARGORG !
DECIMAL





SCR #201
( LP TINPUT APP TCOLD) HEX

: LP  ( b --- )  5 SWAP SYSCALL ;

: TINPUT ( --- n )  0 0 ." ? " QUERY 1 TWORD CONVERT DROP DROP ;

: APP  ( --- ) HEX
 ." To send an ASCII character to the printer," CR
 ." type a HEX number and (cr) after '?' (FF to terminate)." CR
 BEGIN TINPUT DUP FF = NOT WHILE SPACE LP REPEAT 0 0 SYSCALL ;

: TCOLD ( --- ) 0 EPRINT !
 INIT-USER UP @ 6 + 30 CMOVE PAGE APP ;

' TCOLD CFA ' TCOLDCFA !
DECIMAL

SCR #202
( No headers, No block verbs; MINIMUM Test; Target Memory Map)
HEX
0 ' MAPFILE !

6000 ' TVRAM !
6000 ' TEM !
TEM ' TBUF1 !
TBUF1 ' TRO !
TRO A - ' TSPO !
100 ' TARGORG !
DECIMAL






SCR #203
( TCOLD) HEX

: TCOLD  0 0 0 0 21 INTCALL ;

' TCOLD CFA ' TCOLDCFA !
DECIMAL











SCR #204
( Headers, No block verbs; Target Memory Map Constants) HEX

1 ' MAPFILE !

6000 ' TVRAM !
6000 ' TEM !
TEM ' TBUF1 !
TBUF1 52 - ' TRO !
TRO A0 - ' TSPO !
100 ' TARGORG !
DECIMAL





```

(Continued on page 78)

DPMA Houston '85,

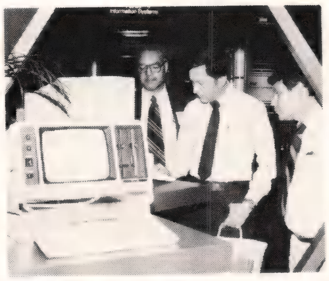
WHERE COMPUTERS GET DOWN TO BUSINESS

Data Processing Management Association's
Annual International Computer Conference and Business Exposition

Oct. 28-30, 1985

Albert Thomas Convention & Exhibit Center
Houston, Texas

Open To All In The Computer or Related Industries
. . .Seminars. . .Workshops. . .Panel Discussions
. . .Tutorials. . .Half-Day Sessions. . .Exhibits



Conference Highlights

- Impact on Information Management
- Small Systems Environment
- Controlling the Revolution
- Back to Basics
- And many more topics
- Conference On-Site Registration Begins on October 27. All Computer Industry Professionals are Welcome to Attend.

Business Exposition

Products, services and supplies by the nation's leading companies including AT&T. . .Digital Equipment Corp.. . .Eastman Kodak. . . IPM. . .Radio Shack. . .Xerox. . .Cincom Systems. . .GE. . .Lanier. . . and many others.

Houston's the perfect site!

As the nation's fourth largest city, Houston has immense financial wealth with expanding opportunities and growth. It also has one of the few remaining center-city areas that is still fun, exciting and comfortable—day and night. The Albert Thomas Convention and Exhibit Center is located in the middle of thriving downtown Houston.

For complete details on DPMA Houston '85, fill out the coupon below.

DPMA HOUSTON '85
WHERE COMPUTERS GET DOWN TO BUSINESS
Oct. 28-30, 1985

Send to DPMA Houston '85
505 Busse Highway
Park Ridge, IL 60068-3191

Please send:
- ☐ Conference registration materials
- ☐ Free exhibit-viewing tickets
- ☐ My company's interested in exhibiting
- ☐ Membership information

Name/Title ____________________
Company ____________________
Address ____________________
City/State-Prov./Zip-Postal Code ____________________
Telephone ____________________

Forth Compiler Listing

(Listing continued, text begins on page 52)

```
SCR #205
( TABORTCFA TNABORT TNUMBER T-FIND)

0 CONSTANT TABORTCFA

: TNABORT ." NOT RECOGNIZED" TABORTCFA EXECUTE ;

: TNUMBER  ( a --- d )
 0 0 ROT DUP 1+ C@ ASCII - = DUP >R + -1 DPL !
 CONVERT DUP C@ BL >
 IF DUP C@ ASCII . = NOT IF TNABORT THEN 0 DPL !
    CONVERT DUP C@ BL > IF TNABORT THEN
 THEN DROP R> IF NEGATE THEN ;

: T-FIND  BL TWORD CONTEXT @ @ (FIND) ;



SCR #206
( TINTERPRET TQUIT TABORT)

: TINTERPRET  ( --- )
 BEGIN T-FIND
    IF DROP CFA EXECUTE
    ELSE HERE TNUMBER DPL @ 1+ NOT IF DROP THEN
    THEN
 AGAIN ;

: TQUIT  ( --- )
 BEGIN CR RP! QUERY TINTERPRET ." OK" AGAIN ;

: TABORT ( --- )
 SP! [COMPILE] FORTH TQUIT ;

' TABORT CFA ' TABORTCFA !


SCR #207
( APP TCOLD)
HEX
: APP CR ." HELLO FROM APP" BEGIN ?TERMINAL ?DUP UNTIL ;

: T.R  ( n1 n2 --- ) >R 0 SWAP OVER DUP D+- <# #S ROT SIGN #>
 R> OVER TYPE ;

: TCOLD ( --- )  0 EPRINT !
 INIT-USER UP @ 6 + 30 CMOVE
 PAGE ." QUIT OK" CR
 INIT-FORTH @ ' FORTH 2+ !
 DECIMAL TABORT HEX T.R ;
DECIMAL

' TCOLD CFA ' TCOLDCFA !



SCR #208
( Block verbs, No headers; BLOCKASC; Target Memory Map) HEX
( Generates DOS-ASCII files of Forth disk block areas)

1 ' MAPFILE !

2 ' T#BUFF !

6000 ' TVRAM !
6000 ' TEM !
TEM 404 T#BUFF * - ' TBUF1 !
TBUF1 52 - ' TRO !
TRO A0 - ' TSPO !
100 ' TARGORG !
DECIMAL



SCR #209
( CHARS BLOCK>FOUT) HEX

: CHARS  ( a --- n )
 DUP C/L + 1-
 BEGIN DUP C@ 20 = OVER 4 TPICK > AND
 WHILE 1-
 REPEAT SWAP - 1+ ;

: BLOCK>FOUT  ( ba fa --- )
 SWAP DUP 400 + SWAP
 DO I CHARS I + I DO I C@ OVER FC! 1 /LOOP
    D OVER FC! A OVER FC! C/L
 /LOOP DROP ;
DECIMAL



SCR #210
( FOUT TINPUT BLOCKASC) HEX

FILE FOUT

: TINPUT ( --- n )  0 0 ." ? " QUERY 1 TWORD CONVERT DROP DROP ;

: BLOCKASC  ( --- )
 CR ." Starting block # (DEC)" TINPUT
 ." # of blocks" TINPUT
 ." DRIVE (0,1)" TINPUT 0 MAX 1 MIN IF DR1 ELSE DR0 THEN
 CR ." DOS file out name" FOUT FILENAME? FOUT FILECREATE DROP
 OVER + SWAP
 DO I BLOCK FOUT BLOCK>FOUT LOOP
 1A FOUT FC! FOUT FILE! DROP FOUT FCLOSE DROP
 CR ." Task complete." CR BYE ;
DECIMAL


SCR #211
( TCOLD) HEX

: TCOLD ( --- ) 0 EPRINT !
 INIT-USER UP @ 6 + 30 CMOVE PAGE
 TBUF1 ' FIRST ! TEM ' LIMIT ! T#BUFF ' #BUFF ! EMPTY-BUFFERS
 CR 1 DENSITY ! FIRST USE ! FIRST PREV !
 DR0  40 ' C/L !
 DECIMAL BLOCKASC ;

' TCOLD CFA ' TCOLDCFA !
DECIMAL






SCR #212
( (TR/W))

: (TR/W)  ( n --- ) USE @ >R ROT USE  ! SWAP MAX-DRV 0
 DO I DR-DEN DENSITY ! DUP BPDRV - -1 >
    IF BPDRV - I 1+ MAX-DRV = IF BYE THEN
    ELSE I DRIVE ! LEAVE
    THEN
 LOOP SPBLK * SPBLK 0
 DO DDUP T&SCALC
```

(Continued on page 81)

THE PROGRAMMER'S SHOP™

helps save time, money and cut frustrations. Compare, evaluate, and find products.

SERVICES

- Programmer's Referral List
- Compare Products
- Help find a Publisher
- Evaluation Literature free
- BULLETIN BOARD - 7 PM to 7 AM 617-826-4086
- Dealer's Inquire
- Newsletter
- Rush Order
- Over 700 products

Free Literature - Compare Products

Evaluate products Compare competitors Learn about new alternatives One *free call* brings information on just about any programming need Ask for any ***"Packet"*** or "Addon Packet" ☐ ADA, Modula ☐ "AI" ☐ BASIC ☐ "C" ☐ COBOL ☐ Editors ☐ FORTH ☐ FORTRAN ☐ PASCAL ☐ UNIX/PC or ☐ Debuggers, Linkers, etc

ARTIFICIAL INTELLIGENCE

ARITY/PROLOG-full, debug, to ASM&C, 16 Meg use, windows, strings. With compiler $1950. MSDOS $495

EXSYS - Expert System building tool. Full RAM, Probability. Why, serious, files PCDOS $275

GC LISP - "COMMON LISP", Help. tutorial, co-routines, compiled functions, thorough. PCDOS Call

LISP-86- "COMMON' subset tutorial, editor, PP, trace. Best to learn. All MSDOS Only $95

M Prolog - full, rich, separate work spaces. MSDOS $725

PROLOG-86 - Learn fast. Standard, tutorials, samples of Natural Language. Exp. Sys. MSDOS Call

TLC LISP - "LISP-machine"-like. all RAM, classes, turtle graph., 8087, Compiler. CPM-86. MSDOS $225

WALTZ LISP - "FRANZ LISP" - like, 611 digits, debugger, large programs. CPM80 MSDOS $159

MicroProlog - improved	MSDOS	$229

BASIC

CADSAM FILE SYSTEM - full ISAM in MBASIC source. MSDOS $150

Quick BASIC by Microsoft - Compiles full syntax of IBM, 640K, BASICA. PCDOS $ 85

BASCOM-86 - Microsoft	8086	279
CB-86 - DRI	CPM86, MSDOS	419
Data Manager - full source	MSDOS	325
InfoREPORTER - multiple	PCDOS	115
Prof. Basic - Interactive, debug	PCDOS	85
TRUE BASIC - ANSI	PCDOS	125

Ask about ISAM, other addons for BASIC

EDITORS FOR PROGRAMMING

BRIEF Programmer's Editor - undo, windows, reconfig. PCDOS Call

EMACS by UniPress - powerful, multifile, windows, DOS, MLISP language built in Source: $949 MSDOS $299

FirsTime by Spruce - Improve productivity. Syntax directed for Pascal ($235) or C ($285).

C Screen with source	86/80	75
Epsilon - like EMACS	PCDOS	195
PMATE - powerful	8086	159
VEDIT - well liked	PCDOS	119
XTC - multitasking	PCDOS	95

COBOL

Microsoft Version II - upgraded. Full Lev. II, native, screens. MSDOS $500

Macintosh COBOL - Full.	MAC	459
MBP - Lev II, native, screen	MSDOS	885
MicroFocus Prof.-full	PCDOS	call
Ryan McFarland-portable	MSDOS	695

C LANGUAGE

C-terp Interpreter by Gimpel, full K&R, .OBJ and ASM interface. 8087 MSDOS $255

INSTANT C - Interactive development - Edit. Source Debug, run. Edit to Run - 3 Secs. MSDOS $395

"INTRODUCING C" - Interactive C to learn fast. 500 page tutorial. examples, graphics PCDOS Call

Q/C 88 by CodeWorks with full **compiler source,** produces decent code, has cross & native MSDOS $295

Wizard C - Lattice C compatible, full sys. III syntax, lint included, fast, lib. source. MSDOS $399

MSDOS: C86-8087, reliable	call
Lattice C - the standard	call
Microsoft C 3.0 - new	259
RUN/C - Interpreter	119
Williams - debugger, fast	call
CPM80: EcoPlus C-faster, SLR	249
BDS C - solid value	125
MACINTOSH: Hippo Level 1	109
Consulair's MAC C with toolkit	299
MegaMax	239

Compare, evaluate consider other Cs

C ADDONS

COMMUNICATIONS by Greenleaf ($149) or Software horizons ($139) includes Modem7, interrupts, etc. Source. Ask for Greenleaf demo.

C SHARP Realtime Toolkit-well supported, thorough, portable, objects, state sys. Source MANY $600

CIndex + -full B+Tree, vari. length field. Source, no royal. MSDOS $359

dbVista FILE SYSTEM - full indexing, plus optional record types, pointers. Source, no royalties. MSDOS $450

Faster C Lattice & C86 users eliminate Link step. Normal 27 seconds. Faster C in 13 sec. MSDOS $ 95

PC Lint - full C program checking and big, small model. All C's. MSDOS $85

CHelper: DIFF, xref, more	86/80	135
CTree - source, no royalties	ALL	345
CURSES by Lattice	PCDOS	110
C Utilities by Essential	MSDOS	149
dBC ISAM by Lattice	8086	219
Greenleaf-200 +, fast.	MSDOS	149
PHACT-up under UNIX, addons	MSDOS	225
ProScreen - windows	PCDOS	275
Windows for C - fast, reliable	MSDOS	175

Call for a catalog, literature, and *solid* value

800-421-8006

THE PROGRAMMER'S SHOP™

128-D Rockland Street, Hanover, MA 02339

Mass: 800-442-8070 or 617-826-7531 1085

RECENT DISCOVERIES

Fortran >> C - FORTRIX creates maintainable translations. MSDOS $995

FORTRAN LANGUAGE

MacFORTRAN - full '77, '66 option. toolbox, debugger, 128K or 512K. ASM-out option MAC $349

RM/Fortran - Full '77. BIG ARRAYS. 8087, optimize, back trace, debug. MSDOS $459

Ask about Microsoft, Supersoft, others.

MS FORTRAN-86 - Improved.	MSDOS	239
DR Fortran-86 - full '77	8086	249
PolyFORTRAN-XREF, Xtract	PCDOS	149

LANGUAGE LIBRARIES

MultiHALO Graphics-Multiple video boards, printers, rich. Animation, engineering business. ANY MS language, Lattice, C86 $195, **for Turbo $95.**

Screen Sculptor - slick, thorough, fast, BASIC, PASCAL. PCDOS $109

GRAPHMATIC - 3D, FTN, PAS	PCDOS	125
File MGNT: BTrieve - all lang.	MSDOS	199
Micro: SubMATH - FORTRAN full	86/80	250
MetaWINDOW - icons, cup	PCDOS	119
PANEL - many lang., terminals	MSDOS	239

OTHER LANGUAGES

Professional Pascal - for performance; extensions like "packages"; "Iterators"; 5 memory models; 64 bit 8087; strings. Space vs. speed optimization options. MSDOS $895

SNOBOL4 + -great for strings, patterns. MSDOS $ 85

MacASM - full, fast, tools	MAC	99
Assembler & Tools - DRI	8086	149
PC FORTH - well liked	MSDOS	89

SUPPORT PRODUCTS

PLINK 86 - a program-independent overlay linker to 32 levels for all MS languages. C86 and Lattice. $299

Multilink - Multitasking	PCDOS	265
Pfinish - Profile by routine	MSDOS	299
Polylibrarian - thorough	MSDOS	85
PolyMAKE	PCDOS	85
ZAP Communications - VT100, TEK 4010 emulation, full xfer.	PCDOS	65

DEBUGGERS

Advanced Trace 86 Symbolic	PCDOS	149
Atron Debugger for Lattice, MSFTN	PCDOS	369
Periscope Debugger -	PCDOS	269
Pfix Plus Debugger	MSDOS	299
TRACE86 debugger ASM	MSDOS	115

Note: All prices subject to change without notice
Mention this ad. Some prices are specials
Ask about COD and POs. All format's available
UNIX is a trademark of Bell Labs

Dr. Dobb's Journal
Telecommunications
Christensen Protocols in C
2400 Baud and Beyond
New Column: Unix Exchange
Queues, Bitmaps, Laserjets and Paranoia
InfoWorld
TECH JOURNAL
BYTE

Dr. Dobb's Stands Apart

Take a good, hard look at the crowded computer magazine market. If you're a serious microcomputerist, one magazine stands apart. ***Dr. Dobb's Journal.***

Dr. Dobb's is not for everyone. It is written by and for expert programmers, and it's the oldest and most technically sophisticated microcomputer publication available. Since 1976, ***Dr. Dobb's Journal*** has been the unchallenged leading source of software tools for advanced programmers.

With the industry moving forward faster than ever, you need to stay a step ahead. ***Dr. Dobb's*** sets the pace with:

- regular columns on C, Unix, MS-DOS and 16-bit software;
- algorithms and problem solving;
- lively discussions of fundamental software issues;
- inside information on commercial languages and operating systems;
- tips on advanced programming topics.

The information and valuable code contained in ***Dr. Dobb's*** makes each issue indispensable for serious computing professionals and enthusiasts. Don't miss a single issue of this valuable resource. Subscribe today. If you aren't fully satisfied, cancel your subscription and keep the first issue as a free sample.

To start your subscription,
fill out this coupon and return it to:

Dr. Dobb's Journal
P.O. Box 27809, San Diego, CA 92128

Yes! Please enter my subscription for 12 issues of:
Dr. Dobb's Journal for $25.

If I am not fully satisfied I will write "cancel" on my subscription invoice and keep the first issue as a free sample.

*Name*____________________

*Address*____________________

*City*____________________

*State*__________ *Zip*__________

☐ *Bill me later* ☐ *I've enclosed $25* *Charge my* ☐ *VISA* ☐ *M/C*

*Card No.*____________________

*Expiration*____________________

*Signature*____________________

Offer good in U.S. only Allow 6 to 12 weeks for delivery. 3046

SOFTWARE TOOLS FOR ADVANCED PROGRAMMERS
Dr. Dobb's Journal

Technical Product Information

FREE

For The Asking

See something you'd like to learn more about? Need more details?

DR. DOBB'S JOURNAL **Reader Service Card**

Name ______________________ Phone ____________

Address ______________________________

City/State/Zip ______________________________

Expiration Date: Jan. 31, 1986 October 1985 #108

Please circle one for each category:

I. My firm or department is a:
A. Systems Integrator/House
B. Software Dev. Firm
C. Hardware OEM or Manuf.
D. DP, MIS or Data Service
E. Consulting Firm
F. Eng. or Science Lab.
G. Other

II. My job function is:
H. Company Mgmt./Admin.
J. Computer Systems Mgt.
K. Programmer/Technical Staff
L. Consultant
M. Engineering Mgmt. or Staff
N. Scientific Mgmt. or Staff
O. Other

III. Number of employees in my firm:
1. Less than 10
2. 10—99
3. 100—499
4. 500—9,999
5. 10,000—or More

IV. This inquiry is for:
P. Immediate Purchase
Q. Future Project

V. Purchasing Authority
(check all that apply)
R. Recommend or Specify
S. Final Decision-Maker
T. No Influence

VI. I advise others about computers, on the average:
6. More Than Once-A-Day
7. Once-A-Day
8. Once-A-Week
9. Not At All

VII. I design/write software professionally
U. Yes
V. No

VIII. I buy computer products through:
(check all that apply)
W. Retail Stores
X. Mail Order Houses
Y. On-site Direct Salespeople
Z. All of the Above

IX. I am currently a subscriber:
A. Yes
B. No

Check each advertisement for corresponding number and circle below:

001	011	021	031	041	051	061
002	012	022	032	042	052	062
003	013	023	033	043	053	063
004	014	024	034	044	054	064
005	015	025	035	045	055	065
006	016	026	036	046	056	066
007	017	027	037	047	057	067
008	018	028	038	048	058	068
009	019	029	039	049	059	069
010	020	030	040	050	060	070
071	081	091	101	111	121	131
072	082	092	102	112	122	132
073	083	093	103	113	123	133
074	084	094	104	114	124	134
075	085	095	105	115	125	135
076	086	096	106	116	126	136
077	087	097	107	117	127	137
078	088	098	108	118	128	138
079	089	099	109	119	129	139
080	090	100	110	120	130	140

					Articles	
141	151	161	171	181	191	201
142	152	162	172	182	192	202
143	153	163	173	183	193	203
144	154	164	174	184	194	204
145	155	165	175	185	195	205
146	156	166	176	186	196	206
147	157	167	177	187	197	207
148	158	168	178	188	198	208
149	159	169	179	189	199	209
150	160	170	180	190	200	210

☐ **Please send me a one year subscription to Dr. Dobb's Journal at $25.00 and bill me later.**

Note:

For quicker, more effective processing of your inquiry, please provide responses to ALL requested information.

Information that's
• Current
• In-depth
• Directed to you on specific products & services

FREE

Send us your POSTAGE-PAID card Today!

DR. DOBB'S JOURNAL **Reader Service Card**

Name ______________________ Phone ____________

Address ______________________________

City/State/Zip ______________________________

Expiration Date: Jan. 31, 1986 October 1985 #108

Please circle one for each category:

I. My firm or department is a:
A. Systems Integrator/House
B. Software Dev. Firm
C. Hardware OEM or Manuf.
D. DP, MIS or Data Service
E. Consulting Firm
F. Eng. or Science Lab.
G. Other

II. My job function is:
H. Company Mgmt./Admin.
J. Computer Systems Mgt.
K. Programmer/Technical Staff
L. Consultant
M. Engineering Mgmt. or Staff
N. Scientific Mgmt. or Staff
O. Other

III. Number of employees in my firm:
1. Less than 10
2. 10—99
3. 100—499
4. 500—9,999
5. 10,000—or More

IV. This inquiry is for:
P. Immediate Purchase
Q. Future Project

V. Purchasing Authority
(check all that apply)
R. Recommend or Specify
S. Final Decision-Maker
T. No Influence

VI. I advise others about computers, on the average:
6. More Than Once-A-Day
7. Once-A-Day
8. Once-A-Week
9. Not At All

VII. I design/write software professionally
U. Yes
V. No

VIII. I buy computer products through:
(check all that apply)
W. Retail Stores
X. Mail Order Houses
Y. On-site Direct Salespeople
Z. All of the Above

IX. I am currently a subscriber:
A. Yes
B. No

Check each advertisement for corresponding number and circle below:

001	011	021	031	041	051	061
002	012	022	032	042	052	062
003	013	023	033	043	053	063
004	014	024	034	044	054	064
005	015	025	035	045	055	065
006	016	026	036	046	056	066
007	017	027	037	047	057	067
008	018	028	038	048	058	068
009	019	029	039	049	059	069
010	020	030	040	050	060	070
071	081	091	101	111	121	131
072	082	092	102	112	122	132
073	083	093	103	113	123	133
074	084	094	104	114	124	134
075	085	095	105	115	125	135
076	086	096	106	116	126	136
077	087	097	107	117	127	137
078	088	098	108	118	128	138
079	089	099	109	119	129	139
080	090	100	110	120	130	140

					Articles	
141	151	161	171	181	191	201
142	152	162	172	182	192	202
143	153	163	173	183	193	203
144	154	164	174	184	194	204
145	155	165	175	185	195	205
146	156	166	176	186	196	206
147	157	167	177	197	197	207
148	158	168	178	188	198	208
149	159	169	179	189	199	209
150	160	170	180	190	200	210

☐ **Please send me a one year subscription to Dr. Dobb's Journal at $25.00 and bill me later.**

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT #27346, PHILADELPHIA, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SOFTWARE TOOLS FOR ADVANCED PROGRAMMERS
Dr. Dobb's Journal
P.O. BOX 13851
PHILADELPHIA, PA 19101

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT #27346, PHILADELPHIA, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SOFTWARE TOOLS FOR ADVANCED PROGRAMMERS
Dr. Dobb's Journal
P.O. BOX 13851
PHILADELPHIA, PA 19101

```
    IF SEC-READ ELSE SEC-WRITE THEN
    1+ 1024  SPBLK / USE +!
LOOP DDROP R> USE ! ;

' (TR/W) CFA 'R/W !



SCR #213
















SCR #214
( Block verbs, No headers; ASCBLOCK; Target Memory Map) HEX
( Generates Forth disk block areas form DOS-ASCII files)

1 ' MAPFILE !

2 ' T#BUFF !

6000 ' TVRAM !
6000 ' TEM !
TEM 404 T#BUFF * - ' TBUF1 !
TBUF1 52 - ' TRO !
TRO A0 - ' TSPO !
100 ' TARGORG !
DECIMAL



SCR #215
( CURBLOCK BCOUNT FIN BC! LINE>BLOCK) HEX

TVARIABLE CURBLOCK  TVARIABLE BCOUNT

FILE FIN

: BC!  ( b --- ) BCOUNT DUP @ 400 =
  IF 1 CURBLOCK +! UPDATE CURBLOCK @ BUFFER DROP 0 OVER ! THEN
  DUP @ DUP 1+ ROT ! PREV @ 2+ + C! ;

: LINE>BLOCK  ( b --- ) DUP D =
  IF 0 ELSE 7F AND BC! 1
```

(Continued on next page)

PRESENTING THE MEGAMAX C COMPILER

FEATURING:
• IN-LINE ASSEMBLY • ONE PASS COMPILATION • SUPPORT OF DYNAMIC OVERLAYS • FULL ACCESS OF MACINTOSH TOOLBOX ROUTINES • AND MUCH MORE...

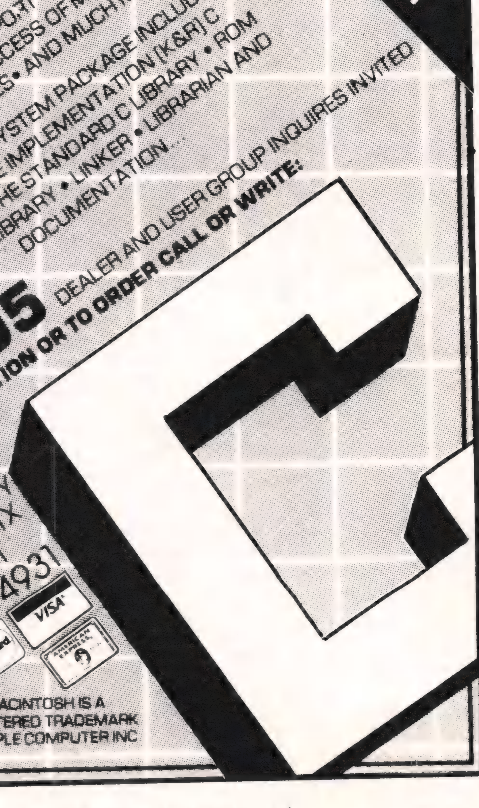
DEVELOPMENT SYSTEM PACKAGE INCLUDES:
• FULL-SCALE IMPLEMENTATION (K&R) C COMPILER • THE STANDARD C LIBRARY • ROM ROUTINES LIBRARY • LINKER • LIBRARIAN AND DOCUMENTATION...

NOW AVAILABLE FOR THE MACINTOSH

$299.95 DEALER AND USER GROUP INQUIRES INVITED

FOR MORE INFORMATION OR TO ORDER CALL OR WRITE:

Megamax, Inc
BOX 851521, DEPT. Y
RICHARDSON, TX
75085-1521
(214) 987-4931



MACINTOSH IS A REGISTERED TRADEMARK OF APPLE COMPUTER INC.

Circle no. **84** on reader service card.

Sidekick for CP/M!

Write-Hand-Man

Desk Accessories for CP/M

NEW! Now with automatic screen refresh!

Suspend CP/M applications such as WordStar, dBase, and SuperCalc, with a single keystroke and look up phone numbers, edit a notepad, make appointments, view files and directories, communicate with other computers, and do simple arithmetic. Return to undisturbed application! All made possible by **Write-Hand-Man.** Ready to run after a simple terminal configuration! No installation required.

Don't be put down by 16 bit computer owners. Now any CP/M 2.2 machine can have the power of *Sidekick.*

Bonus! User extendable! Add your own applications.

$49.95 plus tax (California residents), shipping included! Volume and dealer discounts.

Available on IBM 8 inch and Northstar 5 inch disks. Other 5 inch formats available with a $5.00 handling charge. CP/M 2.2 required; CP/M 3 not supported.

COD or checks ok, no credit cards
Poor Person Software
3721 Starr King Circle
Palo Alto, CA 94306
tel 415-493-3735

Write-Hand-Man trademark of **Poor Person Software,** CP/M trademark of Digital Research, *Sidekick* trademark of Borland International, dBase trademark of Ashton-Tate, WordStar trademark of Micropro, SuperCalc a trademark of Sorcim.

Circle no. **71** on reader service card.

Forth Compiler Listing (Listing continued, text begins on page 52)

```
    BEGIN FIN FC@ 7F AND DUP D = NOT
    WHILE BC! 1+
    REPEAT DROP FIN FC@ DROP C/L SWAP -
 THEN BEGIN DUP WHILE 1- 20 BC! REPEAT DROP ; DECIMAL
```

```
SCR #216
( TYPEFILENAME TINPUT ASCBLOCK) HEX
: TYPEFILENAME  ( fa --- )
DUP FCB 1+ 8 TYPE ." ." FCB 9 + 3 TYPE ;

: TINPUT ( --- n )  0 0 ." ? " QUERY 1 TWORD CONVERT DROP DROP ;

: ASCBLOCK  ( --- )
CR ." DOS in file name" FIN FILENAME? FIN FOPEN
IF CR FIN TYPEFILENAME ."  can't be opened." BYE THEN
CR ." Starting block # (DEC)" TINPUT ." Drive # (0,1)" TINPUT
0 MAX 1 MIN IF DR1 ELSE DR0 THEN
DUP CURBLOCK ! 0 BCOUNT ! BUFFER DROP
BEGIN FIN FC@ DUP 1A = NOT WHILE LINE>BLOCK REPEAT DROP
BEGIN BCOUNT @ 400 < WHILE 20 BC! REPEAT
UPDATE SAVE-BUFFERS FIN FCLOSE DROP
CR ." Task complete." CR BYE ; DECIMAL
```

```
SCR #217
( TCOLD) HEX

: TCOLD ( --- ) 0 EPRINT !
INIT-USER UP @ 6 + 30 CMOVE PAGE
TBUF1 ' FIRST ! TEM ' LIMIT ! T#BUFF ' #BUFF ! EMPTY-BUFFERS
CR 1 DENSITY ! FIRST USE ! FIRST PREV !
DR0  40 ' C/L !
DECIMAL ASCBLOCK ;

' TCOLD CFA ' TCOLDCFA !
DECIMAL
```

```
SCR #120
( Implementation specific constants for CP/M) HEX

INIT-FORTH 12 - CONSTANT FORTHORG   ( Host origin)
148 CONSTANT PATCHMAX   ( end of patch area)
110 CONSTANT TCOLD1     ( address for patch for jump to cold)
116 CONSTANT U8         ( address for patch for user#8)
118 CONSTANT LTIB       ( address for patch for Term-in-buffer)
CD  CONSTANT DOESJMP    ( first byte of run-time create-does))
3   CONSTANT DJMPLEN    ( # of bytes for run-time c-d jmp)
DECIMAL
```

```
SCR #121
( MS-DOS Implementation specific addresses) HEX

INIT-FORTH 1B - ' FORTHORG !
151 ' PATCHMAX !
119 ' TCOLD1 !                                  ( MS-DOS)
11F ' U8 !                                      ( MS-DOS)
121 ' LTIB !                                    ( MS-DOS)
B8  ' DOESJMP !                                 ( MS-DOS)
5   ' DJMPLEN !                                 ( MS-DOS)
DECIMAL
```

```
SCR #122
( VARIABLES) HEX

VARIABLE TVRAMOFF  0 TVRAMOFF !
VARIABLE DOCODE      0 DOCODE !
VARIABLE VERBS#      0 VERBS# !
VARIABLE TVERBS#    0 TVERBS# !
VARIABLE DOWN          0 DOWN !
VARIABLE FIXLINK?  0 FIXLINK? !
VARIABLE !FIX          0 !FIX !
VARIABLE DOVAR?      0 DOVAR? !
DECIMAL
```

```
SCR #123
( CONSTANTS see Examples and Glossary) HEX
0 CONSTANT MAPFILE     ( These constants are loaded by the user)
0 CONSTANT T#BUFF
0 CONSTANT TVRAM
0 CONSTANT TEM
0 CONSTANT TBUF1
0 CONSTANT TR0
0 CONSTANT TSP0
0 CONSTANT TARGORG
0 CONSTANT TCOLDCFA

0 CONSTANT ROOF    ( These constants are initialized by TARGET)
0 CONSTANT NLIST
0 CONSTANT TLIST
0 CONSTANT HEADERS
0 CONSTANT TREETAGCFA DECIMAL
```

```
SCR #124
( CONSTANTS for detection of special cases)
' (+LOOP)  CFA CONSTANT (+LOOP)CFA
' LIT      CFA CONSTANT LITCFA
' (.")     CFA CONSTANT (.")CFA
' (ABORT") CFA CONSTANT (ABORT")CFA
' EXIT     CFA CONSTANT EXITCFA
' 0BRANCH  CFA CONSTANT 0BRANCHCFA
' (LOOP)   CFA CONSTANT (LOOP)CFA
' (/LOOP)  CFA CONSTANT (/LOOP)CFA
' (;CODE)  CFA CONSTANT (;CODE)CFA
' BRANCH   CFA CONSTANT BRANCHCFA
' :      CFA @ CONSTANT DOCOL
' SP0    CFA @ CONSTANT DOUSE
' 0      CFA @ CONSTANT DOCON
' PREV   CFA @ CONSTANT DOVAR
' FORTH  CFA @ CONSTANT DOVOC
```

```
SCR #125
( STRING PUT$ from A.Winfield "The Complete FORTH" Sigma 1983)
HEX
: STRING  ( n --- )
 CREATE DUP C, 0 C, ALLOT
 DOES) 2+ DUP 1- C@ ;

: PUT$  ( a n --- )
 DROP 1-
 DUP 1- C@
 24 WORD DROP
 HERE C@ (
 IF ." String too big" DROP QUIT THEN
 HERE DUP C@ 1+
 ROT SWAP CMOVE ;
DECIMAL

```

```
SCR #126
( TVRALLOT TVARIABLE TPICK TROLL ;S FLIP) HEX

: TVRALLOT  ( n --- )  TVRAMOFF +! ;

: TVARIABLE  ( --- )  TVRAMOFF DUP @ 2 ROT +! TVRAM + CONSTANT ;

: TPICK  ( n1 --- n2 )  2* SP@ + @ ;

: TROLL  ( n --- )  1+ DUP TPICK SWAP 2* SP@ +
 BEGIN DUP 2- @ OVER ! 2- SP@ OVER U( NOT UNTIL DDROP ;

: ;S  ( --- ) 0 'STREAM ! ; DECIMAL

;S   HEX
: FLIP  ( n --- n )
 0 100 U/MOD SWAP 100 * + ; DECIMAL
```

```
SCR #127
( EAKER CASE STATEMENT FROM CEE 9/80 FORTH DIMENSIONS II/3 P 37)

: CASE   ?COMP CSP @ SP@ CSP ! 4 ; IMMEDIATE

: OF   4 ?PAIRS COMPILE OVER COMPILE = COMPILE OBRANCH HERE 0
      , COMPILE DROP 5 ;  IMMEDIATE

: ENDOF  5 ?PAIRS COMPILE BRANCH HERE 0 , SWAP 2
        [COMPILE] THEN 4 ; IMMEDIATE

: ENDCASE  4 ?PAIRS COMPILE DROP BEGIN SP@ CSP @ = 0=
           WHILE 2   [COMPILE] THEN REPEAT CSP ! ; IMMEDIATE

: ASCII  BL WORD  1+ C@  [COMPILE] LITERAL ; IMMEDIATE


```

```
SCR #128
( F$ M$ FILE IMG TARG MAP DMA FCB LB) HEX

B STRING F$ F$ PUT$ TARGET  TMP$
3 STRING M$ M$ PUT$ MAP$

: FILE  ( --- ) CREATE HERE A2 ALLOT A2 0 FILL ;

FILE IMG
FILE TARG
FILE MAP

: DMA  ( fa --- a ) 1+ ;
: FCB  ( fa --- a ) 81 + ;
: LB  ( n --- b ) FF AND ;
DECIMAL

```

(Continued on next page)

Shell Oil
Canadian Government
General Motors
American Airlines
JCPenney
Eastern Airlines
Department of Defense

THESE GIANTS KNOW THE SECRET OF PROGRAMMING PRODUCTIVITY . . FOR $59.95 . . . BE A GIANT!

Dow Chemical
Veterans Administration
Hewlett-Packard
Chase-Manhattan Bank
Coca Cola
Johnson's Wax
Exxon

COMP COMPUTING STANDARD MUMPS WILL AT LEAST DOUBLE YOUR PRODUCTIVITY

CCSM Offers These Advantges:

Full Screen Editor
Virtual Memory (Routines & Variables May Be As Large As A Disk)
Multi-User - Up to 12
B-Tree File Structure
8087 and BCD Support Included
Meets or Exceeds 1984 ANSI Standards
Transportable From Micro to Mini to Mainframe

SPECIAL OFFER
"Cookbook of MUMPS & Diskette"
200 Pages of Programmers Techniques & Routines
ONLY $9.95 - * Save $15.00!
Offer Ends 12/31/85

CCSM 1 - (800) 257-8052 Continental U.S.

	Qty.	
CCSM Single User System	____	$59.95
Graphics Add-On	____	49.95
"Cookbook of MUMPS & Diskette"	____	24.95*
Multi-User System	____	450.00

(Add $2.00 to each order for shipping)

***When Purchased With CCSM**

Guidance Software Inc.
P.O. Box 5362
Kingwood, TX 77325
(713) 360-3561

NAME ____________________
COMPANY ____________________
STREET ____________________
CITY ____________________
STATE ____________ ZIP ____________
PHONE ____________________

TOTAL AMOUNT DUE
$ ____________

☐ Check Enclosed
☐ Credit Card
☐ Amex ☐ Visa ☐ MC
Credit Card Number ______
Exp. Date ______

MS—DOS & PC—DOS are registered trademarks of Microsoft Corp. and IBM Corp. respectively

Circle **no. 6** on reader service card.

Forth Compiler Listing

(Listing continued, text begins on page 52)

```
SCR #129
( T'STREAM TWORD FILENAME?) HEX
: T'STREAM  ( --- a )  TIB @ >IN @ + ;

: TWORD  ( b --- a )  T'STREAM SWAP ENCLOSE DDUP >
 IF DDROP DDROP 0 HERE !
 ELSE >IN +! OVER - DUP >R HERE C! + HERE 1+ R> 1+ CMOVE
 THEN HERE ;

: FILENAME?  ( fa --- )
 ." ? " QUERY
 TIB @ 1+ C@ ASCII : =                       ( default or select?)
 IF  3A TWORD 1+ C@ 40 - 0 MAX 0F MIN OVER FCB C! ( select disk)
 ELSE 0 OVER FCB C! THEN                            ( default disk)
 DUP FCB 0C + 15 0 FILL DUP FCB 1+ 0B BLANK            ( clear FCB)
 2E TWORD COUNT 8 MIN 3 TPICK FCB 1+ SWAP CMOVE         ( load FCB)
 BL TWORD COUNT 3 MIN 3 TROLL FCB 9 + SWAP CMOVE ; DECIMAL

SCR #130
( ISFF FSEARCH FNSEARCH FDELETE FCREATE) HEX

: ISFF  ( f --- f )  FF = IF FFFF ELSE 0 THEN ;

: ISZERO  ( f --- f )  0= IF 0 ELSE FFFF THEN ;

: FSEARCH  ( fa --- f )  FCB 11 SWAP SYSCALL LB ISFF ;

: FNSEARCH ( fa --- f )  FCB 12 SWAP SYSCALL LB ISFF ;

: FDELETE  ( fa --- f )  FCB 13 SWAP SYSCALL LB ISFF ;

: FCREATE  ( fa --- f )
 DUP 81 0 FILL FCB 16 SWAP SYSCALL LB ISFF ;
DECIMAL


SCR #131
( FOPEN FCLOSE SETDMA FREAD FWRITE) HEX

: FOPEN ( fa --- f )
  DUP 80 SWAP C! FCB F SWAP SYSCALL LB ISFF ;

: FCLOSE  ( fa --- f )
 FCB 10 SWAP SYSCALL LB ISFF ;

: SETDMA  ( fa --- )
 DMA 1A SWAP SYSCALL DROP ;

: FREAD  ( fa --- f )
 FCB 14 SWAP SYSCALL LB ISZERO ;

: FWRITE  ( fa --- f )
 FCB 15 SWAP SYSCALL LB ISZERO ; DECIMAL

SCR #132
( SETDMA FREAD FWRITE FILE@ FILE! FILEEOF) HEX

: FILE@  ( fa --- f )
 DUP SETDMA DUP FREAD IF FFFF FF ELSE 0 0 THEN ROT C! ;
```

THE SYMBOLIC *SINE QUA NON*
L I S P

THE fifth generation language
P R O L O G

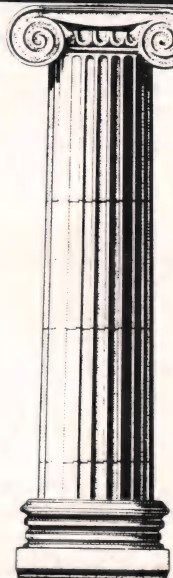
Cybermetrics **UNXLISP-86** (tm) Features:

- 1 Megabyte address space.
- LAMBDA, NLAMBDA, MACRO with displacement.
- CATCH, THROW, ERRSET, & ERR.
- SAVE/RESTORE virtual images to/from disk.
- &OPTIONAL, &REST, and &AUX parameters.
- Tree-structured object list, functional directories.
- Lisp utilities with source, including pretty-printer, structure editor, debugging functions.
- Works with any text editor through exec function.
- Extensive manual – written in ENGLISH
- Requires 320K memory and runs under MSDOS and PCDOS versions 2.0 and later.

TECHNICAL
(408) 725-1344

$49.00

ADD $3.00 FOR
SHIPPING & HANDLING



Implementing the full Edinburgh Syntax as described by Clocksin and Mellish.

Recognized by Japan as providing unparalleled opportunity for artificial intelligence.

Applications:

- The highest level of a hierarchial robotic control system.
- Machine recognition of natural language.
- Expert systems and knowledge engineering.

Optional:

- Virtual memory
- Special libraries
- Language extensions
- Large model

Requires 192K memory and runs under MSDOS and PCDOS versions 2.0 and later.

TECHNICAL
(215) 646-4894

educational package
$29.95
other versions $50–$500

automata design assoc.
1570 ARRAN WAY
DRESHER, PA 19025
TO ORDER CALL **(215) 646-4894** USE IT FOR 30 DAYS

VISA
MASTERCARD

Circle **no. 12** on reader service card.

```

: FILE! ( fa --- f ) DUP C@
IF DUP SETDMA DUP FWRITE SWAP 81 0 FILL ELSE DROP 0 THEN ;

: FILEEOF ( fa --- f )
C@ FF = IF FFFF ELSE 0 THEN ;

: FILECREATE  ( fa --- f )
DUP FSEARCH NOT IF DUP FDELETE DROP THEN FCREATE ;
DECIMAL


SCR #133
( FC@ FC! IMG TARG MAP F$ M$) HEX

: FC@  ( fa --- b )
DUP C@ 80 = IF DUP FILE@ DROP THEN
DUP C@ 1+ DUP 3 TPICK C! + C@ ;

: FC!  ( b fa --- )
DUP C@ 80 = IF DUP FILE! DROP THEN
DUP C@ 1+ DUP 3 TPICK C! + C! ;
DECIMAL






SCR #134
( SPACEFILE SPACESFILE TYPEFILE U.RFILE) HEX
: SPACEFILE  ( fa --- ) 20 SWAP FC! ;

: SPACESFILE  ( n fa --- )  SWAP 0 MAX ?DUP
IF 0 DO DUP SPACEFILE LOOP THEN DROP ;

: CRFILE  ( fa --- )  D OVER FC! A SWAP FC! ;

: TYPEFILE  ( a n fa --- )  ROT ROT DUP 0>
IF OVER + SWAP DO I C@ 7F AND OVER FC! 1 /LOOP
ELSE DDROP THEN DROP ;

: U.RFILE  ( # r fa --- )  ROT ROT
0 SWAP >R SWAP OVER DUP D+-
<# #S ROT SIGN #> R> OVER -
4 ROLL DUP ROT SWAP SPACESFILE TYPEFILE ; DECIMAL


SCR #135
( ID.FILE MAXNFA) HEX

: ID.FILE  ( nfa fa --- )  SWAP COUNT 1F AND ROT TYPEFILE ;

: MAXNFA  ( nfa1 a nfa2 --- nfa3 a ) [ SMUDGE ]
BEGIN DUP DUP 0= SWAP @ A081 = OR NOT
WHILE DUP PFA DUP CFA @ DOVOC = SWAP 4 + @ 0= NOT AND
   IF ROT ROT 3 PICK PFA 2+ @ MAXNFA ROT THEN
   OVER OVER U< NOT
   IF ROT DDUP U< NOT IF DROP DUP THEN ROT ROT THEN
   PFA 4 - @
```

(Continued on next page)

WorkArea

A Text Management & Presentation System *for Computer Product Vendors*

- WorkArea incorporates your word-processor (or the full function editor provided) into an object-oriented data-base containing text, numbers, strings, vectors, calculations, scripts, windows, interrogatives and more.
- WorkArea can be used as an authoring system, for building minor expert or knowledge-base systems.
- WorkArea can build on-line and interactive —
  - — product descriptions: answer specific questions;
  - — tutorials & user-guides: get users started;
  - — diagnostics: assist users in isolating problems;
  - — documentation (on-line or printed): with user-guide, reference, tutorial and help-system consolidated into a co-ordinated package.
- WorkArea is available in read-only versions.
- WorkArea runs on the IBM PC and compatibles with 256 RAM, DOS 2.xx, floppy or fixed-disk, monochrome or color monitor.

Complete WorkArea package: $275

Demo and documentation package: $30

(site license or special orders possible)

DMCQ

Systems Consulting and Software Development

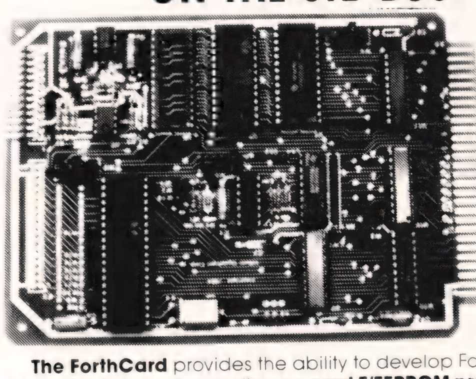
To order: P.O. Box 602 (707) 743-1907
Ukiah, CA 95482

Send a check or money order. Sorry, we do NOT accept phone, credit card or COD orders. Please supply a street address for UPS delivery.

Circle **no. 21** on reader service card.

ForthCard

ON THE STD-BUS



as Low as $179

The ForthCard provides the ability to develop Forth code using a single board, including **on-card E/EEPROM programming.** Operating as a **stand alone computer** or as an STD bus CPU, this card is perfect for projects that require a simple dedicated SBC, while allowing for memory and I/O expansion. Forth programs can be entered directly via CRT or downloaded from your PC's serial port, blown into optional non-volatile RAM or EEPROM, and auto-executed on power-up.

The evaluation unit is available for as little as **$299. OEM** Versions of the card may be purchased for **$179** each. (Qty. 10).

The **ForthBox** is a complete STD bus oriented system including the ForthCard, Disk Controller, Disk Drive(s), STD card cage Cabinet and Power Supply.

CALL TODAY FOR COMPLETE INFORMATION

HiTech Equipment Corporation

9560 Black Mountain Road, San Diego, CA 92126

FOR IMMEDIATE ACTION CALL

619-566-1892

MasterCard VISA

Circle **no. 23** on reader service card.

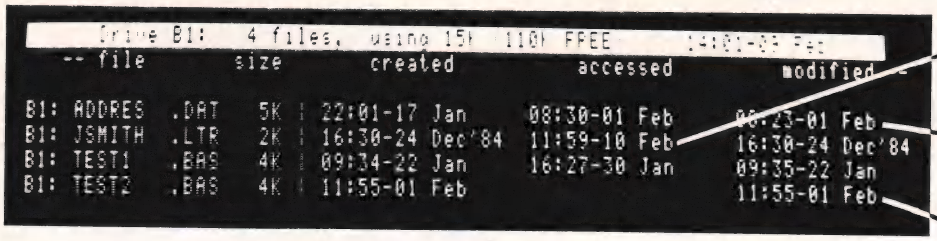
```
 REPEAT DROP [ SMUDGE ] ;
DECIMAL



```

```
SCR #136
 ( NFTAG VERBTAG SFLAG >TTAG) HEX

 : NFTAG ( nfa --- )
 DUP C@ 20 OR SWAP C! ;

 : VERBTAG ( cfa --- )
 DUP 2- @ (;CODE)CFA =
 IF 0 SWAP [COMPILE] FORTH CONTEXT @ @ MAXNFA DROP
 ELSE 2+ NFA
 THEN NFTAG ;

 : SFLAG  ( f --- f ) SWAP 1- SWAP ;

 : >TTAG  ( --- ) TREETAGCFA EXECUTE ;
DECIMAL

```

```
SCR #137
 ( PFSKIPS)

 : PFSKIPS ( a1  --- a1 / a2 0 )
 DUP @
 CASE  LITCFA OF LITCFA >TTAG 2+ 0 ENDOF
      (.")CFA OF (.")CFA >TTAG 2+ COUNT + 2- 0 ENDOF
      (ABORT")CFA OF (ABORT")CFA >TTAG 2+ COUNT + 2- 0 ENDOF
      0BRANCHCFA OF 0BRANCHCFA >TTAG 2+ 0 ENDOF
      BRANCHCFA OF BRANCHCFA >TTAG 2+ 0 ENDOF
      (LOOP)CFA OF (LOOP)CFA >TTAG 2+ 0 ENDOF
      (/LOOP)CFA OF (/LOOP)CFA >TTAG 2+ 0 ENDOF
      (+LOOP)CFA OF (+LOOP)CFA >TTAG (LOOP)CFA >TTAG 2+ 0 ENDOF
 ENDCASE ;



```

```
SCR #138
 ( CREATE-DOES)) HEX

 : CREATE-DOES)  ( cfa --- )
 @ DUP VERBTAG DUP C@ DOESJMP =
 IF ' DOES) CFA VERBTAG DJMPLEN 2- +
    BEGIN 2+ DUP @ EXITCFA = NOT
    WHILE PFSKIPS ?DUP IF DUP @ >TTAG THEN
    REPEAT
 ELSE DUP DUP HEX CR U. 2+ NFA ID.
      ." ---Code verb assumed." CR DECIMAL
 THEN DROP ;
DECIMAL

```

DATESTAMPER™ has the answers



```
   Drive B1:    4 files,  using 15k  110k FREE        14:01-03 Feb
  -- file         size        created            accessed          modified

B1: ADDRES  .DAT   5K | 22:01-17 Jan      08:30-01 Feb      08:23-01 Feb
B1: JSMITH  .LTR   2K | 16:30-24 Dec'84   11:59-10 Feb      16:30-24 Dec'84
B1: TEST1   .BAS   4K | 09:34-22 Jan      16:27-30 Jan      09:35-22 Jan
B1: TEST2   .BAS   4K | 11:55-01 Feb                        11:55-01 Feb
```

When did we print that letter?

Has the mailing list been updated?

Which is the latest version?

DateStamper™ keeps your CP/M computer up-to-date!

- avoid erasing the wrong file
- back-up files by date and time
- keep dated tax log of computer use
- simplify disk housekeeping chores

OPERATION: DateStamper extends CP/M 2.2 to automatically record date and time a file is created, read or modified. DateStamper reads the exact time from the real-time clock, if you have one; otherwise, it records the order in which you use files. Disks prepared for datestamping are fully compatible with standard CP/M.
INSTALLATION: Default (relative-clock) mode is automatic. Configurable for any real-time clock, with pre-assembled code supplied for all popular models. Loads automatically at power-on.
UTILITIES: Enhanced SuperDirectory • Powerful, all-function DATSWEEP file-management program with date and time tagging • Installation and configuration utilities
PERFORMANCE: Automatic. Efficient. Versatile. Compatible.

Requires CP/M 2.2. Uses less than 1K memory. Real-time clock is optional.

When ordering please specify format

8" SSSD, Kaypro, Osborne Formats	**$49**
For other formats (sorry, no 96 TPI) add $5.	
Shipping and handling	**$3**
California residents add **6%** sales tax	

MasterCard and Visa accepted

Specialized versions of this and other software available for the Kaypro.
CP/M is a registered trademark of Digital Research, Inc.

Write or call for further information

Plu*Perfect Systems

BOX 1494 • IDYLLWILD, CA 92349 • 714-659-4432

Circle **no. 69** on reader service card.

```
SCR #139
( TREETAG) HEX
: TREETAG  ( cfa --- )  [ SMUDGE ]
DUP 2+ NFA C@ 20 AND NOT
IF 0 SWAP DUP 2+ DOCODE ! DUP VERBTAG DUP @
  CASE DOCODE @ OF SFLAG ENDOF
    DOCOL OF
      BEGIN 2+ DUP @ DUP EXITCFA = SWAP <;CODE>CFA = OR NOT
      WHILE PFSKIPS ?DUP IF DUP @ TREETAG THEN
      REPEAT SFLAG ENDOF
    DOUSE OF DUP 2+ @ DUP 14 > SWAP 36 < AND
      IF DUP EXECUTE @ TREETAG THEN SFLAG ENDOF
    DOVAR OF SFLAG ENDOF
    DOCON OF SFLAG ENDOF
  ENDCASE SWAP NOT
  IF DUP CREATE-DOES> THEN
THEN DROP [ SMUDGE ] ; DECIMAL


SCR #140
( TFIX;CODE X TFIXNUL APPTAG) HEX
: TFIX;CODE  ( --- )
' CREATE CFA VERBTAG ' CONSTANT CFA VERBTAG
' : CFA VERBTAG ' USER CFA VERBTAG ' EXIT CFA VERBTAG
HEADERS IF ' BYE CFA TREETAG THEN ;

: X R> DROP ;
: TFIXNULL ( --- )
' X CFA TREETAG  80E0 ' X NFA ! ;        ( patch for null)

: APPTAG  ( --- )  CR CR ." Start tagging."
TFIX;CODE TCOLDCFA TREETAG
HEADERS ?DUP IF TREETAG TFIXNULL THEN
."    Finished tagging." ; DECIMAL

' TREETAG CFA ' TREETAGCFA !


SCR #141
( HITKEY DDRIVE LOADFCB IMAGESAVE) HEX

: HITKEY  ( --- )  ." and hit any key. --- " KEY DROP ;

: DDRIVE  ( --- )
CR CR ." Insert DOS disk in default drive," HITKEY ;

: LOADFCB  ( a n fcba --- )
DUP 21 0 FILL 1+ SWAP CMOVE ;

: IMAGESAVE  ( --- ) DDRIVE
F$ IMG FCB LOADFCB
IMG FILECREATE DROP
HERE FORTHORG DO I 1- SETDMA IMG FWRITE DROP 80 /LOOP
IMG FCLOSE DROP CR CR ." Step 1 complete." DDRIVE BYE ;
DECIMAL


SCR #142
( FRAMEIN RAZE) HEX

: FRAMEIN  ( --- )
```

(Continued on next page)

The Journal of Forth Application and Research

The aim of the *Journal* is to provide a reliable source of state of the art techniques and applications of Forth to scientific and industrial problems. The *Journal's* editorial review board is drawn from the foremost workers in Forth in the U.S., Europe and Canada. Past issues had in-depth coverage of such topics as Robotics, Data Structures, Forth Computers, Real-Time Systems, and Extended Address Computing. The *Journal* is open to all new work in Forth as well as the entire range of threaded interpretive languages and Forth-like systems.

The *Journal* is beginning its third year of publication. Published quarterly, the *Journal* contains applications and techniques papers, technical notes, review papers, algorithms, book reviews, and conference abstracts.

1985 Rochester Forth Conference Proceedings
The Proceedings of the 1985 Rochester Forth Conference will appear as a Special Issue of Volume 3 of the *Journal*. The conference had a theme of software management and engineering and how to improve software productivity. Invited papers discuss a space shuttle experiment using Forth, the automation of an airport with Forth, and Forth in the development of MAGIC/L. (Also available as a single issue for $20.; outside North America, $25.)

Subscriptions Volume 3, 1985 — Corporations and Institutions, $100.; Individual subscribers, $40. Please add $20 for airmail delivery outside N. America, and make checks in US funds from a US bank or international money order, payable to the *Journal of Forth Application and Research*, P.O. Box 27686, Rochester, NY 14627.

Circle **no. 61** on reader service card.

VANCE info systems is pleased to announce
THE MOST COMPLETE C FUNCTION LIBRARY AVAILABLE TO DATE!

C. lib

C FUNCTION LIBRARY

*"**C lib**" is the most functional library available for software written in C, providing over 200 routines, extending the capabilities of C on the IBM PC. The library is available under the DeSmet (C Ware) C88 compiler, will soon be available in MicroSoft C, Lattice C and other C compilers, and runs using MS Dos 1.1 and later versions.*

Featuring:
- ***Unix/Xenix Compatibility***
- ***Windowing Library***
- ***Math Functions***
- ***String Functions***
- ***Asychronous Buffered, Interrupt-driven Communications***
- ***8087/MicroSoft Floating Point conversions***
- ***And much more***

Documentation is offered in an easy to use printed manual or on disk for personal printing needs, complete with programming examples and follow-up demo programs.

*The "**C lib**" **C FUNCTION LIBRARY** is offered at only **$195** less than most available today.*

*For further information on "**C lib**" please call or write us.*

VANCE info systems

2818 clay street • san francisco, california 94115 • (415)922-6539

IBM is a trademark of International Business Machines Corp. C88 is a trademark of Computer Innovations, Inc. Lattice is a trademark of Lattice Inc. Xenix, MicroSoft C and MS DOS are trademarks of MicroSoft Inc. Unix is a trademark of Bell Labs. Inc. "C lib" is a trademark of VANCE info systems.

Circle **no. 154** on reader service card.

```
SP@ 2- 2- ' ROOF !
VERBS# @ 2+ 0 DO 0 LOOP
SP@ ' NLIST !
HERE ROOF 2+ ! FFFF NLIST !
TVERBS# @ 3 + 2 * 0 DO 0 LOOP
SP@ ' TLIST !
FORTHORG DUP TLIST !
TARGORG - TLIST 2+ !
HERE NLIST 8 - !
FFFF NLIST 4 - ! ;

: RAZE  ( --- )
VERBS# @ 2+ TVERBS# @ 3 + 2 * + 0 DO DROP LOOP ; DECIMAL


SCR #143
( COUNTVERBS NFALISTFILL) HEX

: COUNTVERBS   ( nfa --- ) [ SMUDGE ]
 BEGIN DUP DUP 0= SWAP @ A081 = OR NOT
 WHILE DUP PFA DUP CFA @ DOVOC = SWAP 4 + @ 0= NOT AND
    IF DUP PFA 2+ @ COUNTVERBS THEN
    DUP C@ 20 AND IF 1 TVERBS# +! THEN PFA 4 - @ 1 VERBS# +!
 REPEAT DROP [ SMUDGE ] ;

: NFALISTFILL   ( nfa --- ) [ SMUDGE ]
 BEGIN DUP DUP 0= SWAP @ A081 = OR NOT
 WHILE DUP PFA DUP CFA @ DOVOC = SWAP 4 + @ 0= NOT AND
    IF DUP PFA 2+ @ NFALISTFILL THEN
    DUP PFA 4 - @ SWAP DOWN @ ! -2 DOWN +!
 REPEAT DROP [ SMUDGE ] ;
DECIMAL


SCR #144
( WINDOWTOP FRAMEBOTTOM) HEX
: WINDOWTOP  ( a1 --- a2 )
 BEGIN DUP DUP @ SWAP 2- @ U< NOT
 WHILE 2-
 REPEAT 2- ;

: FRAMEBOTTOM  ( a1 n --- a2 ) SWAP
 BEGIN DUP @ 3 PICK U< NOT
 WHILE 2-
 REPEAT SWAP DROP ;

: WINDOWBOTTOM  ( a1 a2 --- a3 ) @ OVER
 BEGIN DUP @ DUP 4 PICK U< NOT SWAP FFFF = NOT AND
 WHILE 2-
 REPEAT SWAP DROP SWAP WINDOWTOP DDUP U<
 IF SWAP THEN DROP ; DECIMAL


SCR #145
( OPENMOVE WINDOWSORT) HEX

: OPENMOVE  ( a1 a2 a3 --- )
DUP DUP 2+ SWAP 5 PICK SWAP - DUP >R PREV @ SWAP BMOVE
3 PICK 2+ SWAP 2+ 3 PICK 5 ROLL - BMOVE
```

WINDOWS FOR C™

Advanced screen management made easy

#1

WINDOWS FOR C™
was ranked #**1** by
PC TECH JOURNAL
(Review by William Hunt, July 1985)

In comparison with five
Windowing Utilities for C
[Window Machine, (Lattice Windows),
CRIOS, Building Blocks II and C-LIB]
WINDOWS FOR C came out as the winner.

Overall Quality	# **1**
Display Speed	# **1**
Ease of Use	# **1**
Code Size	# **1**
Documentation	# **1**

For all popular C-compilers operating under PCDOS. Plus versions for XENIX & UNIX.

Trademarks — Lattice Inc.; UNIX, AT&T; XENIX, Microsoft

```
R@ - 2+ PREV @ SWAP R> BMOVE ;

: WINDOWSORT  ( a --- )
BEGIN DUP WINDOWTOP DUP @ FFFF = NOT
WHILE OVER OVER @ FRAMEBOTTOM
   DDUP WINDOWBOTTOM OPENMOVE
REPEAT DROP DROP FORTHORG NLIST ! ;
DECIMAL



SCR #146
( FINDTAG OFFSETS) HEX

: FINDTAG  ( a1 --- a2 )
BEGIN 2+ DUP @ C@ 20 AND UNTIL ;

: OFFSETS  ( --- ) CR CR ." Calculating offsets."
NLIST 2+ @ TLIST NLIST TVERBS# @ 0
DO FINDTAG DUP >R @ HEADERS NOT IF PFA CFA THEN
   ROT - SWAP 4 + SWAP OVER 2- @ + OVER
   2+ ! R@ @ HEADERS NOT IF PFA CFA THEN OVER !
   R@ 2+ @ SWAP R>
LOOP
DROP DUP 2+ @ SWAP 6 + ! DROP
."   Offsets calculated." ;
DECIMAL


SCR #147
( VERBSLIST FIXSMUDGE) HEX

: VERBLISTS  ( --- ) 0 VERBS# ! 0 TVERBS# !
CR CR ." Creating verb lists."
[COMPILE] FORTH CONTEXT @ @ COUNTVERBS FRAMEIN
ROOF DOWN ! CONTEXT @ @ NFALISTFILL
1 BUFFER DROP ."   Sorting." ROOF WINDOWSORT
DECIMAL CR CR VERBS# @ U. ." (DEC) Verbs in host Forth."
CR TVERBS# @ U. ." (DEC) Verbs in target code."
OFFSETS ;

: FIXSMUDGE  ( --- )
TLIST 4 + TVERBS# @ 0
DO DUP I 4 * + @ HEADERS NOT IF 2+ NFA THEN
   DUP C@ DF AND SWAP C!
LOOP DROP ; DECIMAL


SCR #148
( NEWLINK TARGMAP DUMP/=) HEX

: NEWLINK  ( nfa1 --- nfa2 )
TLIST 2- 2-
BEGIN 2+ 2+ DDUP @ = UNTIL
2- 2- @ SWAP DROP 0 FIXLINK? ! ;

: TARGMAP  ( n nfa --- )
SWAP TARGORG + DUP 4 HEX U.R SPACE MAPFILE
IF 4 MAP U.RFILE MAP SPACEFILE DUP MAP ID.FILE MAP CRFILE
ELSE DROP THEN OUT @ SWAP ID. OUT @ - A + DECIMAL
BEGIN 1- DUP 0< NOT WHILE SPACE REPEAT DROP
50 OUT @ F + < IF CR 0 OUT ! THEN ;

: DUMP/=  ( a --- ) CR 20 - 50 DUMP TARG FILE! DROP
TARG FCLOSE DROP IMG FCLOSE DROP FIXSMUDGE ABORT ; DECIMAL
```

(Continued on next page)

WINDOWS FOR DATA™

Featuring One-Step Data Entry◆

Now you can code fast, powerful data entry windows, improve user convenience – reduce input errors.

All the power, convenience and flexibility of the **#1** window utility for the IBM PC. Our WINDOWS FOR C™ combined with a professional window-based data entry system.

Complete control over screen display and entry of data within a convenient flexible window environment.

	WINDOWS FOR C	WINDOWS FOR DATA (Includes WINDOWS FOR C)
PCDOS	**$ 195**	**$ 295**
PC/XENIX	**$ 395**	**$ 595**
UNIX	**CALL**	**CALL**

WINDOWS FOR DATA™ provides versatile, easy-to-use data entry functions that operate within windows.

CAPABILITIES INCLUDE:

- Pop-up data entry windows
- Multiple field types
- Data validation functions
- Field-specific & context-sensitive help
- Lotus-style menu design
- Single field entry option
- Date, time and string utilities
- Dynamic control of data-entry environment

◆ User input to data-structure variables without intervening code.

Vermont Creative Software
21 Elm Ave.
Richford, VT 05476
802-848-7738, ext. 31

Full source available. Master Card & Visa accepted. Shipping $ 2,50. VT residents add 4% tax.

Circle no. **27** on reader service card.

(Listing continued, text begins on page 52)

```
SCR #149
( PATCHADD VAR) HEX

: PATCHADD  ( a1 --- a2 f ) DUP
 CASE TCOLD1     OF   TCOLDCFA 1 ENDOF
    INIT-USER    OF   TSPO     0 ENDOF
    U8           OF   TRO      0 ENDOF
    LTIB         OF   TSPO     0 ENDOF
    UP           OF   TRO      0 ENDOF
    RPP          OF   TRO      0 ENDOF
    DUP          OF   DUP @    1 ENDOF
 ENDCASE ROT DROP ;

: VAR  ( a --- )  DOVAR? @ NOT OVER RPP = NOT AND
 IF CR ." Images not matched at " DUP HEX 4 U.R DECIMAL DUMP/=
 THEN PATCHADD DROP DUP FF AND TARG FC! FLIP FF AND TARG FC! ;
DECIMAL


SCR #150
( NEXTNFA REPAIRADD) HEX

: NEXTNFA  ( nfa1 --- nfa2 )
 NLIST 2- BEGIN 2+ DDUP @ U< UNTIL @  SWAP DROP ;

: REPAIRADD  ( a1 --- a2 )
 DUP @ DOVAR = IF 2 DOVAR? ! THEN
 DUP PATCHMAX < IF PATCHADD ELSE @ 1 THEN
 IF FIXLINK? @ ?DUP IF SWAP DROP NEWLINK THEN
    TLIST
    BEGIN DDUP @ U< NOT
    WHILE 2+ 2+
    REPEAT 2- @ -
 THEN ;
DECIMAL



SCR #151
( VERB>TARG) HEX

: VERB>TARG  ( nfa --- )
 DUP NEXTNFA SWAP
 DO IMG FC@ DUP I C@ =
    IF I FIXLINK? @ =
        IF DF AND !FIX @ IF 0 FIXLINK? ! 0 !FIX ! THEN
        THEN TARG FC! 1
    ELSE IMG FC@ 100 * + 201 - I @ =
        IF I REPAIRADD DUP FF AND TARG FC!
            FLIP FF AND TARG FC! 2
        ELSE I VAR 2
        THEN
    THEN DOVAR? @ IF -1 DOVAR? +! THEN
 /LOOP ;
DECIMAL


SCR #152
( DISCARD RELOCATE) HEX

: DISCARD  ( n --- )
```

WIZARD C

LANGUAGE FEATURES

- Full K & R standard syntax
- enum, void, unsigned char, unsigned long supported
- Structure assignment
- Non-unique structure members
- Full Lint checking included

CODE OPTIONS

- 6 Memory models up to 1M Byte of Code & Data
- 8087/80186/80286 support
- In-line assembler using C symbols
- ROMable code

LIBRARY FEATURES:

- K & R Standard I/O supported
- DOS 2.0 & 3.0 interfaces
- Transcendental math library
- IBM ROM BIOS interface supported
- Library source code included

Only $450.

(617) 641-2379

AMERICAN EXPRESS

MasterCard

VISA

Fast execution speed: Sieve of Eratosthenes 6 seconds with 2 register variables on a PC

VAX/VMS, UNIX Cross-compilers Available

WIZARD systems software, inc.

11 willow court, arlington, ma 02174

Circle **no. 116** on reader service card.

```
 ?DUP IF IMG SWAP 0 DO DUP FC@ DROP LOOP DROP THEN ;

: RELOCATE  ( --- n )  0 DOVAR? !
 0 OUT ! 0 !FIX ! 0 FIXLINK? !
 CR CR TARGORG 4 HEX U.R ."  HEX (target origin)" CR CR
 MAPFILE IF TARGORG 4 MAP U.RFILE MAP CRFILE THEN
 FORTHORG 0 NLIST 8 - TLIST
 DO I TLIST = NOT IF DUP I @ TARGMAP I @ FIXLINK? ! THEN
    I @ [ ' ! NFA ] LITERAL = IF 1 !FIX ! THEN
    SWAP I @ SWAP - DISCARD
    I @ VERB)TARG I @ DUP NEXTNFA DUP ROT - ROT + 4
 /LOOP SWAP DROP DECIMAL ;
DECIMAL
```

```
SCR #153
( VERB)TARGNH) HEX

: VERB)TARGNH  ( cfa --- )
 DUP NEXTNFA SWAP
 DO IMG FC@ DUP 1 C@ =
    IF TARG FC! 1
    ELSE IMG FC@ 100 * + 201 - I @ =
        IF I REPAIRADD DUP FF AND TARG FC!
            FLIP FF AND TARG FC! 2
        ELSE I VAR 2
        THEN
    THEN DOVAR? @ IF -1 DOVAR? +! THEN
 /LOOP ;
DECIMAL
```

```
SCR #154
( RELOCATENH) HEX

: RELOCATENH  ( --- n )
 0 OUT ! 0 FIXLINK? ! 0 DOVAR? !
 CR CR TARGORG 4 HEX U.R ."  HEX (target origin)" CR CR
 MAPFILE IF TARGORG 4 MAP U.RFILE MAP CRFILE THEN
 FORTHORG 0 NLIST 8 - TLIST
 DO I TLIST = NOT IF DUP I @ 2+ NFA TARGMAP THEN
    SWAP I @ SWAP - DISCARD
    I @ VERB)TARGNH
    I @ DUP NEXTNFA
    DUP ROT - ROT + 4
 /LOOP SWAP DROP DECIMAL ;
DECIMAL
```

```
SCR #155
( SETUPDOS CLEANUPDOS) HEX

: SETUPDOS  ( --- )
 DDRIVE F$ IMG FCB LOADFCB
 BEGIN IMG FSEARCH
 WHILE CR ." TARGET.TMP not found on default drive" DDRIVE
 REPEAT IMG FOPEN DROP
 CR CR ." Target code filename"
 TARG FILENAME? TARG FILECREATE DROP MAPFILE
 IF TARG FCB MAP FCB 21 CMOVE
    M$ MAP FCB 9 + SWAP CMOVE MAP FILECREATE DROP
 THEN ;
```

(Continued on next page)

File Edit Aux

dBASE.com

Now CP/M programs the MACINTOSH way!

The available library of time-tested CP/M-80 programs is so vast that almost all popular computers, including IBMs and Apples, have been modified so they can run them.

Now these same programs run the Macintosh way. Easily. Without expensive or complicated software. Without hardware changes. With C.P.Mac.

With C.P.Mac you can open and start CP/M programs the same ways you open paintings or other Macintosh documents. Graphic controls and cues will help you while they are running — including a continuous display of disk availability! And you can make copies of your CP/M files for other Macintosh programs, or vice versa.

C.P.Mac works great with just one disk drive. It also works great with up to three, including hard disks and RAM disks.

- Choose from countless entertainment, business, and educational programs.
- Special-purpose programs for accounting, agriculture, architecture, construction, legal, medical/dental, real estate, retail, wholesale, and a host of other professions.
- Move your current CP/M activities to the Macintosh without skipping a beat. Spreadsheets, data bases, whatever. It's easy. There's no reprogramming or retraining — you save time and money.
- C.P.Mac will move any file from any CP/M machine to your Mac. It's easy with your Imagewriter cable and CP/M programs you already have — no confusing modem transfers.

A unique and quality Macintosh program at a value price of $135.00. Call and order now at (714) 953-8985 or mail to: **LOGIQUE**
30100 Town Center Dr. "O" Suite 198
Laguna Niguel, CA 92677

MC, Visa, C.O.D. or contact your dealer.

Visit us in September at APPLEFEST at the Long Beach Convention Center, Booth 1400.

8080 emulation with about 42k TPA on a 128k Macintosh. Z80 emulation with 63k TPA on a 512k Macintosh or XL. CP/M 2.2 BDOS and BIOS calls. LST and PUN output to the printer. RDR and CON input from the keyboard, and CON outputs to the C.P.Mac window. ADM-3A emulation with reverse video and line insert/delete.
Macintosh is a trademark licensed to Apple Computer, Inc. CP/M is a registered trademark of Digital Research, Inc. C.P.Mac is a trademark of Logique.

0 hard disk sys-cp/mac 100
63K

Circle **no. 9** on reader service card.

```

: CLEANUPDOS  ( --- ) IMG FCLOSE DROP
IMG FDELETE DROP TARG FILE! DROP TARG FCLOSE DROP MAPFILE
IF 1A MAP FC! MAP FILE! DROP MAP FCLOSE DROP THEN ; DECIMAL


SCR #156
( TARGETMEMORY)

: TARGETMEMORY  ( n --- ) CR CR
." (HEX) Target Memory Map" HEX
CR TVRAM TVRAMOFF @ + 1- 4 U.R ."  Last needed byte of RAM."
CR TVRAM 4 U.R ."  Target variable RAM start {TVRAM}."
CR TEM 4 U.R ."  Target-end-of-memory {TEM}."
CR TBUF1 4 U.R ."  Start of block buffer area {TBUF1}."
CR TRO 4 U.R ."  Initial return stack pointer {TRO}."
CR TSPO 4 U.R ."  Initial stack pointer {TSPO}."
CR TARGORG + 1- 4 U.R ."  Target code end address {last used}."
CR TARGORG 4 U.R ."  Target origin {TARGORG}."
DECIMAL ;




SCR #157
( TARGET) HEX

: TARGET  ( pfa or 0 --- )
DUP IF CFA THEN ' HEADERS !
TCOLDCFA 2+ NFA FENCE ! FREEZE
APPTAG
FORTHORG 301 = IF IMAGESAVE THEN
VERBLISTS
SETUPDOS
HEADERS IF RELOCATE ELSE RELOCATENH THEN
CR CR HEX DUP U. DECIMAL ."  (HEX) Bytes in target. "
TARGETMEMORY CLEANUPDOS FIXSMUDGE RAZE
CR CR ." Target compilation complete." ;
DECIMAL


```

End Listing

The LIBRARIAN Inc.

SOURCE INCLUDED

has taken the drudgery out of generating plots. These libraries contain the most complete collection of plotting subroutines you can buy. Each collection places over 50 routines (in FORTRAN or C language) at your fingertips. The library makes it easy to:

- plot semi-log and log-log graphs
- draw arcs and circles
- scale charts to exact sizes
- label text in different sizes, directions, and orientations
- design bar charts and pie charts
- and much more

Operating Systems:	**MS-DOS and PC-DOS**
Plotters Supported:	**Houston Instruments** **Hewlett-Packard** **HPGL compatibles**
Compilers Supported: (specify compiler when ordering)	**Microsoft FORTRAN** **DR FORTRAN-77** **RM FORTRAN** **CI-C86** **MWC86**
Price:	**$295.00** Visa, MC, American Express
For More Information:	**The Librarian, Inc.** **10435 Greenbough, #110** **Stafford, TX 77477** **(713) 499-7662 or 491-2088**

Circle **no. 148** on reader service card.

CP/M™

UPGRADES - FROM $49.95!

ConIX Operating System: Adds more UNIX™-like features to any 48K+ CP/M-80 or compatible micro then ever before. It's a total upgrade for CP/M! Includes I/O Redirection and Pipes (uses memory or disk), perfected User Areas, Command and Overlay Path Searching, Auto Screen Paging, 8Mb Print Buffering, 22 new SysCalls, Function Keys, "Virtual" disk system, Archiver (saves over 50% disk), extensive command language, 300+ variables, 100+ commands, and even more! Uses 1/2K TPA, 0-27K disk minimum. 60-second install time!

ConIX Library I - XCC Utilities: Over 20 utilities written in the ConIX shell language, including hierarchical directories, interactive debugger, move/copy/link multiple files, print files with pagination, review disk files for deletion, unerase disk with stats, full-screen TYPE, and more. Source code included!

"(ConIX) goes a great deal farther in flexibility than any other CP/M shells. . . this system has all of its competitors beat."
As reviewed in Computer Language, June 1985.

- ConIX Basic System: List $99.95, *Special - only $49.95*
- ConIX Programming System: List $99.95, *Special - only $49.95*
- Both Systems 100% Complete: List $165, *Special - only $79.95*
- ConIX Pull-Down Menu: List $35, *FREE with any ConIX system*
- ConIX Library I: List $49.95, *with any ConIX system only $34.95*

Prices include manual, 8" disk, and user support. 5¼" conversions 48TPI $6, other $10. Shipping: $4.50 UPS, $10 Canada, $15 overseas. NY residents add sales tax. Dealers wanted - *special incentives!*

Computer Helper Industries Inc.
P.O. Box 680 Parkchester Station, NY 10462
800-628-2828 X513 24hr. order hotline - COD only
212-652-1786 M-F:10-6 sales/technical staff

UNIX: AT&T Bell Labs, CP/M: Digital Research, ConIX: Computer Helper Ind.

Circle **no. 22** on reader service card.

ATTENTION

C·PROGRAMMERS

File System Utility Libraries

All products are written entirely in K& RC. Source code included, No Royalties, Powerful & Portable.

BTree Library **75.00**

- High speed **random** and **sequential** access.
- **Multiple keys** per data file with up to **16 million** records per file.
- **Duplicate keys**, variable length data records.

ISAM Driver **40.00**

- Greatly speeds application development.
- Combines ease of use of **database manager** with flexibility of **programming language.**
- Supports **multi key files** and **dynamic index definition.**
- Very **easy** to use.

Make **59.00**

- Patterned after the **UNIX** utility.
- Works for programs written in **every language.**
- **Full macros, File name expansion** and **built in rules.**

Full Documentation and Example Programs Included.

For more information call or write:

softfocus

1343 Stanbury Drive
Oakville, Ontario, Canada
L6L 2J5
(416) 825-0903
(416) 844-2610

Credit cards accepted.

Dealer inquiries invited.

Circle **no. 120** on reader service card.

Fatten Your Mac for $5.00

Thanks to Macintosh owners everywhere, *Dr. Dobb's* January 1985 issue #99 was a runaway best-seller.

Now, due to popular demand, the Doctor has reprinted the sought-after **Fatten Your Mac** article from the sold-out January issue. The article explains how you can pack a full 512K of memory into your system, and save half the cost by performing the upgrade yourself.

To order: Enclose $5.00 for each copy with this coupon and send to:

Dr. Dobb's Journal, 2464 Embarcadero Way, Palo Alto, CA 94303
Outside U.S., add $2.00 per copy for shipping & handling.

Please send me _____ copies of **Fatten Your Mac. ALL REPRINT ORDERS MUST BE PREPAID.**

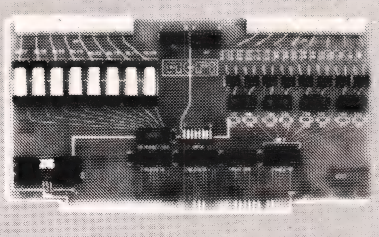
Name ____________________

Address ____________________

City __________ State _____ Zip _____

Please allow 6–9 weeks for delivery.
Offer expires Dec. 31, 1985 **3108**

MULLEN S-100: Real Time, Real World CONTROLLER

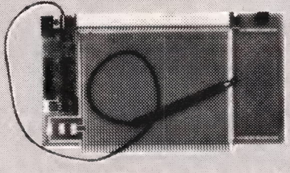


ICB-10 CONTROLLER BOARD
$219, assembled and tested.

This 8 channel digital I/O controller can monitor inputs and control outputs. It features an easy to read manual that has schematics, component list, and programming examples as well as provocative insights on potential applications.

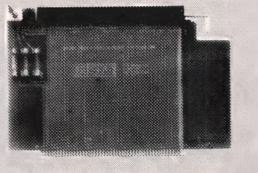
Examples of applications are included in a reprinted article that demonstrates two MULLEN CONTROLLER BOARDS in an interactive system that: feeds a cat; irrigates a garden dependent on soil moisture; closes the window when it rains; controls the thermostat for optimum comfort; controls appliances, lights, security system, and weather monitoring station (logging temperature, wind speed and direction, and graphing pollution content of the atmosphere.) Solenoids, microswitches, pneumatic actuators, pH sensors, and other devices are used in this system.

MULLEN S-100: DEBUGGERS



TB-4a EXTENDER BOARD
The latest in our TB line, the most widely used add-ons in the industry. Features logic probe, formed-lead edge connectors, pulse catcher switch and reset button.

$110 assembled and tested.



ZB-1 ZIF EXTENDER BOARD
This debugger features Zero Insertion Force edge connectors for easy board changes and long life. Expect 2,000 or more insertions rather than the usual 300 to 400 with tension type connectors.
$159, assembled and tested.

MCPI
MULLEN COMPUTER PRODUCTS, INC.

AVAILABLE:
Priority One Electronics,
Chatworth, CA • (213) 709-5111
Jade Computer Products
Hawthorne, CA • (213) 973-7707
E²I Computer Products
Hayward, CA • (415) 786-9203

Mullen Computer Products is the industrial distributer for **CompuPro®**'s products. For more information, call us at (415) 783-2866 or write MCPI, 2260 American Ave., #1, Hayward, CA 94545. OEM sales available from factory. Prices are subject to change without notice.

Circle **no. 104** on reader service card.

PC BRAND
We're NEW in this magazine but an Old Standby for IBM Program Developers

IF YOU PROGRAM IN 'C' DON'T TURN THE PAGE!

PADLOCK

RECOMMENDED!

Protect Any Program Using Anyone's Diskettes

You are a program developer seeking a low-cost method to protect against piracy? You are with a prominent company fearful of embarrassing lawsuits should employees copy programs illegally?

We have a devilishly clever solution for the two of you we call PadLock™. It gives you the keys to padlock any program you want—yours or someone else's—without buying a stockpile of expensive fingerprinted disks.

PadLock is software. One of its programs formats any manufacturer's disk, embedding your secret code in sectors the operating system will ignore. Another program installs this code in any program files you name, whether YourProg or WordStar™, or any .COM or .EXE files. From then on there's no picking the lock without the combination: the coded software together with a coded disk.

It's not despised copy prevention. Protected programs can be copied anywhere: to hard disks, to backup diskettes. It's a lock and key scheme. A protected program will just lock up at load time without a keyed diskette (such as its own) in one of the drives. The key disk may then be removed to free the drive.

Padlock is economical. You buy software that fingerprints 100 disks and their programs for $2.50 each, not costly fingerprinted disks you may never use at maybe $6.00 each.

Tell you what. PadLock keeps track of how many disks it has protected. Try it out. Format up to 5 test disks. If PadLock doesn't suit you, return it within 30 days for full refund.

Incidentally, don't look elsewhere. PadLock is available only from PC Brand.

Code	Our Price:
M0100 100 formats	$250

BRIEF

OUTSTANDING!

Is Anything But. A Whopper of an Editor

Dvorak in *Infoworld* said it for us: "the word is getting around...that Brief, The Programmer's Editor is simply the best text editor you can buy." With a name that belies its thoroughness, Brief™ has every feature you've ever contemplated for your editor-in-chief. Here goes:

Text, from keyboard or files, is housed in multiple buffers, and scrolled through one or more windows you open, close, resize. A text buffer may be called to different windows to view two areas at once. A change in one changes both. Text blocks may be marked for printing, writing to files, movement to scrap buffers for cut and paste into other buffers, or deletion, with as many "undo" levels as you want.

To find and fix, Brief has text search abilities rivaling "grep," with wild cards for matching, indifference to intervening characters, acceptance of character ranges, even multiple choice of patterns and their replacements.

If you use Lattice, C86™, or Wizard, and have 320k, you can compile your C program without ever leaving Brief. In fact, Brief will then find the lines with errors, and march you through the text for repairs.

On to macros! Parts of Brief were written with its own Lisp-like macro language. It has structure, 32-character variable names, conditional execution, loops, and you can actually read it! Nothing like the hieroglyphs we've seen elsewhere. A compiler comes with your copy, and prewritten macros in source to give you the flavor. Among the notables: one automatically indents C text; another turns Brief into a word processer with margin setting and wraparound.

There's a bulletin board with public domain macros contributed by a mushrooming cult of macromaniacs, and we've even hearing of megamoney prices for magnificance.

Then there's ... but, really, we're out of space. Who named this product, anyway?

Product Code: U0590
List Price: $195
Needs 192k

Our Price: $CALL

FIRSTIME

Assistant Programmer for Hire. Real Cheap.

Not interested in interpreters? Then explore the other fast lane to coding: language-specific editors. Among them, FirsTime™ is the strongest C editor we have seen, with many more commands and built-in utilities.

FirsTime asks what program structure you want and sets it up for you. It could be "main", or "function", or "if-else" or "while" or any component of C. Ask for it and FirsTime fills in its skeleton, tending to semi-colons and curly braces.

It puts placeholders into these structures to prompt you to enter workable code—the initializers, conditionals and counters which govern an "if", for example. It checks that variables you enter have been declared, and highlights errors until fixed. Material can be pulled into ten buffers for movement elsewhere. Your last delete is held in one of them for "undoing". There's search and replace, writing of any marked block to a satellite file or the printer.

FirsTime can even transform one type of structure into another, such as "for" into "while" or "while" into "do".

Movement through the screen highlights the successive logical blocks of your program so that its structure is always apparent. And FirsTime can hide different levels of code detail to show you only the outer framework of your program.

FirsTime saves keystrokes to add speed, indeed, but more important, it lets you think at a higher level and makes a big contribution toward error-free code.

Product Code: E0460
List Price: $295

Our Price: $229

LMK

A Unix-like "Make" Makes Light of System Building in Any Language

If you have ever built a complex system, you know the time loss and tedium of recompiling, rebuilding libraries and relinking modules because a snippet or two of code has changed. Batch files are no answer. You need batches of them to avoid redoing everything indiscriminately.

Instead, imagine making a change deep in a system, and simply telling Lattice's LMK™ to take over. No further thinking or keystrokes. LMK will rebuild your final product, however involved and complex, by doing just what is needed and no more.

How? You write a command file which expresses, bottom to top, all the elements comprising your system and all its dependencies: what gets compiled to make what object file using what options; what is built into libraries; what is linked into the final EXE file. Through the life of your system LMK keeps track of the last time every action was performed. Run LMK and, tracking each branch, it looks only for elements which changed later than a dependent element further along the branch, using date and time information in the file directory. Any source file newer than its object file, for example. Only those elements and their dependents are re-made. All other instructions are bypassed.

LMK does not care what programming language you use; it's not just for C. For that matter, LMK can apply to more than programming. It can be used for *any* set of tasks which can be accomplished through commands issued to the operating system.

Wherever your imagination leads you, LMK will find the shortest path to get the job done. Minimum time, minimum effort software.

Product Code: L2100
List Price: $195

Our Price: $159

CODE SIFTER

Find the Fast Track for Your Program

You know it runs, but how fast? If it can't win, better not enter your program in today's horse race. Don't let it out of the stable until you've put it through its paces with our value- priced stopwatch.

Code Sifter finds the sluggish spots in your program. On its own, it will divide a COM or EXE file of any size into thirty-two equal partitions. Or specify the partition boundaries with addresses, or with symbols, if your linker has produced a symbol map. Then tell Sifter to run your program. It samples your object file at precisely timed intervals and counts how many times it finds the instruction pointer in each partition. Job done, it reports the number and percentage of hits in each.

You are in for some surprises when you discover how unbalanced the activity is likely to be, and that's why Code Sifter is so valuable. It profiles just where you can best spend time optimizing your code, or even converting to assembler subroutines.

Code Sifter has a number of monitoring options. You can tell it to include any combination of your program, DOS, and BIOS in its analysis. You can specify the sampling rate. You can tell Code Sifter the number of times to run a program, and between each run discard the less active ranges, and re-partition the hot spots, so that you zoom in ever-tighter on small areas of code. Right down to the last byte! Try it on the sample program that comes with your disk.

Code Sifter. It will give you the racer's edge.

Product Code: N3100
List Price: $119

Our Price: $ 99

C-SPRITE

Lattice's Own Symbolic Debugger for Lattice® C

This versatile companion to your compiler gives you the best of both worlds for an out of this world price. Hand it a COM or EXE file produced by the Lattice C™ compiler (using the -d option) and C-Sprite will speak your language: your function names, your variable names, your data types, and the line numbers from your source code. At the same time you can get a close-up view of machine addresses and machine-coded instructions, if you want to scrutinize just what machinations the compiler (or an assembler) contrived.

You already know how to converse with C-Sprite™ if you are familiar with Microsoft's Debug. Lattice began with that well-known command language, and added to it considerably: You can work with data in hex, as you might expect, but you can also differentiate between C's data types to cause the debugger to treat addresses as strings, long integers, even pointers, etc., both in display and entry.

C-Sprite can set breakpoints using symbols or addresses. You can submit clusters of commands to be executed at the breakpoints, or set commands that execute until a condition is met.

C-Sprite even has macros—use your source code variable names in a macro to dump the contents of entire C structures, for example. And you can debug through one of the COM ports with a second terminal so as not to disturb your program's display screen. What's more, if you link with Plink86™, C-Sprite can even tackle overlays.

Program doctors will find plenty of implements to rummage through in this kitbag.

Product Code: L2300
List Price: $175

Our Price: $145

CURSES

A Screen Management Interface to Swear By

Curses manages the screen of the IBM PC in the same fashion as the curses utility of Unix and similar operating systems. Use it to adapt programs which call Unix's curses functions for screen management, and need the equivalent library when moved to the PC for re-compilation. Or use it when creating software on the PC to assure that it is Unix compatible.

Curses is a library of eighty-four functions and macros which can keep any number of screen images in memory. A screen may be full or partial size, and any can be summoned to the physical screen at your programs's command.

Within a screen, Curses employs a vast function set to get characters, wrap lines, scroll, blank lines, highlight— virtually any conceivable tool needed to update the screen. The product supports color, and all four memory models. Its input functions give you control over whether to echo each character to a memory screen. In keeping with the terminal orientation of Unix curses, the physical screen is repainted (at high speed) only when your program calls a refresh function.

Writing screen management code leads to unspeakable snarls and expressions. Swear off! Let Curses clean up your language.

Code:	List Price:	Our Price:
L0850	$125	$100
L0860 with Source	$250	$200

EXPORT EXPERTISE

PC BRAND ships anywhere. We'll prepare the export documents and ship to you or your agent by air parcel post, air freight or courier. Phone or Telex your order for speed. To pay by credit card, please provide number, date of expiry, name and billing address of card. Or wire funds to PC BRAND, c/o Chemical Bank, 126 East 86th St., New York, N.Y. 10028, Account No: 034-016058. We will ship immediately and acknowledge by Telex if a number provided.

CRAFTSMAN™ QUALITY SOFTWARE FOR 'C' & OTHER PROGRAMMERS

C-TREE

NEW FEATURES!

B-Tree File Manager, Source Code, No Royalties!

A b-tree can be infested with bugs, so before buying one, ask its age. In a stand of saplings, this one is a real cequoia. C-tree™ has been around since 1979. (It became Digital Research's Access Manager™). That means seasoned, sturdy code which hasn't cracked under prolonged and widespread use.

C-tree comes in C source code, revealing all you've ever wanted to know about how b-trees are written. Provided you bind it into your binary application, you can re-distribute c-tree without royalties.

And if all this is disappointing, now the good part. C-tree's design splits nodes to allow any number of users to access an index file simultaneously even when updates are in progress so that multi-user configurations and adaptation to networks are possible.

The latest version has new features: support of variable record length data files; multiple key indexes in a single physical file; MS-DOS and Unix record locking examples.

Thanks to source code which does not deviate from K&R, C-tree can travel. Binary has always meant finding a substitute file manager for yet another compiler, operating system, or computer; then changing all the function calls program-wide; then starting the whole testing process anew. That's over. Tests in many environments prove that C-tree gives your application a ticket to anywhere.

C-tree permits any number of keys for a data file, supports duplicate keys, alphanumeric or numeric, etc., etc.: it's a big product with everything you'd expect. Beyond that it is intelligently designed as both a high level set of ISAM routines to minimize your coding by handling all details of adding a record on its own, for example; and as low level operations which you can access directly. Either way C-tree maintains optimal index structures which will find a record amongst a million ten byte keys in no more than five disk seeks.

Product Code: F0660(	Our Price:
List Price: $395	$329

TEXT TOOLBOX

Tackles Text Tangles

Unix™ boasts a number of muscular utilities that are migrating to the PC world. Lattice has assembled a cluster of the most useful text management tools into a single package.

"Grep" looks for text patterns in any number of files. Want all occurrences of a global variable throughout a program system? Want to search all programs in a directory, down paths to other directories, or all files on a disk? Need to find all the function calls in an entire program system? Grep can do it with a powerful expression syntax that goes far beyond your text editor's search command.

"Ed" is similar to the well-known Unix editor. It offers search and replace with "grep's" syntax, block move, read and write, optional line numbering, append, insert, delete, and this unusual facility: you can instruct "ed" to apply a file of commands to any number of target files, even complicated changes and text additions, such as those created by "diff".

"Diff" compares text files line for line. Its output is a precise list of instructions telling what to do to make two files the same, a list which can be handed to "ed" to do.

Code:	List Price:	Our Price:
L2200	$120	$100
L2205 with Source	$240	$200

CVUE

Make Your Own Editor

CVUE is a low-priced screen-oriented text editor which does most of the things that a good editor should do, and boasts full DOS 2.0 directory path name support in reading and writing files.

It was written by the Lattice programmers who felt forgotten by the folks who write WP software. They needed easy entry of non-display characters such as control codes and escape sequences, not footnotes; indenting and undenting, not italics; pattern searching, not spell checking. So CVUE™ was born.

CVUE only supports in-memory text files, but with memory at today's prices, creating and maintaining files of over 500 K is practical. As compensation, CVUE is very compact and fast. It actually runs in computers with only 64 K of memory and uses no tediously slow overlays.

The power of CVUE is its ease of customization. And when you take advantage of the Source Code option, the resultant editor can be made truly your own.

Code:	List Price:	Our Price:
L2240	$75	$69
L2245 with Source	$200	$220

USED COMPILERS WELCOME

Trade In for the Latest Model Lattice C

Has your compiler run out of gas? Has your model been discontinued? Is it falling behind for lack of new parts?

Even if yours is in good shape, you have surely noticed there are more options and accessories produced to run with the Lattice C compiler than any other. Don't do without these additives any longer. It's time for new license plates. Trade in your original disks and manual of any of the compilers below and we'll send you Lattice's most up-to-date model.

From then on you will be adopted by Lattice for full, direct support by their technical specialists.

	Price:
Microsoft MS-DOS/PC-DOS C	$150
Computer Innovations C86, Mark Williams C, Digital Research C, Whitesmith's C	$200

PRE-C

Thorough "Lint"-like Analysis Now on the PC

Unix users long for a "lint" to give programs a thorough cleaning before they disappear into a compiler.

Pre-C™ looms larger than "lint." It finds problems your compiler won't. Problems that a debugger will have trouble figuring out. Even problems which will cause trouble with other compilers.

Pre-C finds all the syntactical tripwires that will blow out a compiler, sure, but it goes after subtler problems: code which will never be accessed, casts with suspect conversions, variables declared as external but never used, functions never called, obsolete usage (even C has changed), machine-dependent expressions with will inhibit portability.

Compilers work with one module at a time. They know nothing of other modules which only meet up at link time. Pre-C can look at *all segments of your program at once* and report to you any inconsistencies of inter- module references: conflicting data type declarations, parameter lists in function calls which disagree with the functions themselves in number or data type, declarations of external functions which differ from their definition.

Pre-C uses the Unix System III compiler standard to safeguard maxium portability anywhere in the C world. There are then plentiful command line options to advise Pre-C what to flag and what to forgive, useful during early coding when some functions are empty or incomplete. The output of each analysis can be filed for use with subsequent Pre-C runs, so work is not performed redundantly.

Pre-C lets you develop standing profiles of binary libraries. In any C program you subsequently write, Pre-C can use these profiles to make sure your calls to those libraries' functions are perfect.

This is a big product which will work miracles in speeding large system development. 128K minimum; 192K recommended.

Product Code: P0590	Our Price:
List Price: $395	$329

30-DAY MONEY-BACK GUARANTEE

We will return the purchase price[a] of any product in this space if you decide it fails to meet your needs except those coded "(" by requirement of their suppliers. We want you to be completely satisfied with your purchase. Go through the manuals, *try out the products*, and make a thoroughly informed decision whether they are right for you.

The only requirement for refund or credit is that we must *receive* your returned product within 30 days of its shipment, and it must in our judgment be in 100% resaleable condition.

[a] Not shipping and handling.

SUPPORT DIRECT FROM LATTICE

Products with codes beginning "L" are all fresh from Lattice Inc., the premier software developer serving the C professional marketplace. PC BRAND has made special arrangements for Lattice to provide telephone support for these products directly.

Take advantage of this unusual opportunity. Buy from PC BRAND, and you get to talk to the experts themselves!

dBC

Switch from dBASE Language to C for Power, Speed

There are a lot of dBASE™ file users out there. Most of them just keep data bases and use dBASE's limited reporting facilities. They're not programmers, so they don't use the dBASE programming language. But they'd like more for their efforts, and that's a business opportunity.

dBC™ links C to dBASE. It creates and maintains files and their indexes which exactly replicate dBASE file design. So dBASE can read and update them. And the reverse. dBC can use any files created by dBASE. Now C and dBASE can operate on the same data bases interchangeably.

That opens up the widespread culture of dBASE installations to exploitation by C programmers. Now you can replace the resident dBASE language with the speed of C. And you no longer have to write every line of code, as in the dBASE language, because now you've unlocked C's vast storehouse of off-the-shelf libraries and utilities.

dBC's functions parallel all dBASE's file handling commands, many of them decomposed to give you closer-in control. The manual discusses each in detail, and demonstration source files on your disk show how every function is used. Use dBC for custom work for clients, or design generalized programs for manipulation and reporting of dBASE data bases.

Or use dBC on its own. It's a complete ISAM file manager for use with the Lattice C compiler whether or not dBASE will ever be used in tandem, has versions for all four memory models, and can have sixteen index and data files open at once.

Code & Version:	List Price:	Our Price:
L00II: Dbase II Compatible	$250	$225
LCCII: with Source Code	$500	$450
L0III: Dbase III Compatible	$250	$225
LCIII: with Source Code	$500	$450

TERMS AND CONDITIONS OF SALE

Licenses: Each price is for a license to use a product on a single computer and does not constitute its ownership. We will inquire for you about availability of multiple machine licenses at a single site. Except for those coded "#", products may be used to create programs for distribution without royalty payments or additional licenses, provided said programs do not substantially replicate the products themselves.
Compatibility: PC BRAND's standard products are designed to operate with the IBM® PC, XT or AT under PC-DOS and require no more than 128K of RAM unless indicated. Non-IBM machines using MS-DOS: If in doubt as to total compatibility, please determine from manufacturer.
Returns: Defective parts will be replaced. Products coded "(" are rendered unreturnable if sealed envelopes containing diskettes are opened. Otherwise, to return a product for refund or credit, call (212) 410-4000 for required prior authorization.
Payment: We honor MasterCard, Visa, American Express (no surcharge), wired funds, checks in advance (in NY State, add sales tax), and will ship COD (U.S. only) for cash, money order, or certified check (no fee). Purchase orders accepted from larger corporations and institutions at our discretion; terms net 30, 2% a month late penalty.
Shipping & Handling: U.S.: UPS Surface: 1st product $6, each add'l $3. UPS 2nd Day Air: 1st product $10, each add'l $4.50. UPS Next Day Air or Federal Express 1-2 Day Air: 1st product $18, each add'l $6. Federal Exp. Next Morning: 1st product $28, each add'l $7. International: Charges depend on destination and shipping method. $10 per shipping container for export or customs forms.

To Order Products or Catalog, Call Us at...

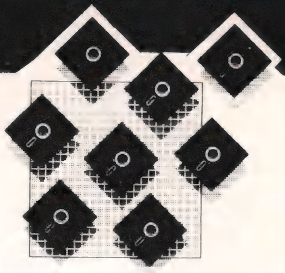
800-PC BRAND

That' (800) 722-7263. In NY State or Outside U.S. call (212) 410-4000.
PC Brand, P.O. Box 474, New York, N.Y. 10028
TELEX: 667962 (SOFT COMM NYK)

Prices, terms, and specifications subject to change without notice.

SAVE $5 Just tell us this code: R10

Circle **no. 108** on reader service card.



Neon, Version 1.0

Company: Kriya Systems, Inc.
505 N. Lake Shore Drive, Suite 5510, Chicago, IL 60611, (312) 822-0624
Computer: Apple Macintosh (128K)
Price: $155.00
Circle Reader Service No. 101
Reviewed by Bruce Horn

This review is reprinted with the permission of the author, Bruce Horn, and The Club Mac News, *in which it originally appeared (Vol. 1, No. 12, May 1985).*

I have not written many programs in Forth. I've always considered Forth to be a sort of high-level assembly language, fun for hacking around in, but not suited to development of large programs. I have written Smalltalk programs, though, and when I heard that Charles Duff and the folks at Kriya had developed Neon, an object-oriented programming environment that was said to be a cross between Smalltalk and Forth, I was intrigued; is this the long-awaited favorite programming environment for the Macintosh?

History of Object-Oriented Programming

Object-oriented programming began in the early 1960's with the development of Simula by the Norwegian Computation Center in Oslo, Norway. Simula was the first language to implement the Class construct. In the early 1970's, the Learning Research Group at Xerox Palo Alto Research Center began implementation of the Smalltalk programming environment. Smalltalk was the first system to be designed completely around the Class/Object concept. Many other languages have since provided support for classes, objects, and subclassing, including CLU, Ada, C++ (an extended version of C), and several versions of LISP, though objects and classes are not as well-integrated into these languages as they are in Smalltalk. The ease of experimentation with objects was one of the key reasons that much of the interesting user interface designs borrowed for Macintosh and Lisa were created in the Smalltalk system. (See *Smalltalk-80: The Language and its Implementation,* by Adele Goldberg and Dave Robson, for more details.) The latest version of Smalltalk has been licensed to several universities and computer companies; Tektronix markets a 68000-based Smalltalk machine for around $14,000, and a version of Smalltalk-80 will soon be available for the Macintosh XL.

Classes and Objects—a Quick Overview

In Neon, classes define objects, a convenient packaging of 1) the data that defines the object and 2) the object's actions. For example, Class **Pen** defines what a pen object looks like (as far as its internal state) and acts like (the messages it responds to). The internal state, or instance variables of a **Pen** object, might include location, direction, and pen size; the messages that pens respond to would include **move:**, **turn:**, **goto:**, and **home:**. (The Neon tutorial has an excellent discussion of the implementation of a **Pen** class, with examples and line-by-line explanation of the source code). Class definitions are bracketed by **:Class** and **;Class** words, and include a list of instance variables (which may themselves be objects) and methods (the procedures run when a message is sent to the object).

Defining a class is like designing a little computer, with its own memory (the instance variables) and instruction set (the messages it can receive). In using a computer, usually you don't concern yourself with how the computer executes the instructions you give it. If you did, and you counted on the various nuances of the implementation of each instruction, programming the computer would be much more complicated. Classes protect you from the need to know about the implementation of the object (computer) by allowing you to talk to the object through a set of messages.

Responses to messages include returning a result on the stack, changing the internal state of the object, sending messages to other objects, or just producing a side-effect (such as drawing a line on the screen). **10 move: Bic** sends the message **move:** to the pen object named **Bic**, changing its location instance variable and possibly drawing a line on the screen; **45 turn: Bic** turns the pen 45 degrees to the right. One advantage of having a pen *object* do your line drawing on the screen instead of just calling Quickdraw directly is that you can have many separate pens, each with its own state, and you can send messages to them independently. To paraphrase Dan Ingalls, one of the developers of Smalltalk, programming with objects is like working with trained animals instead of pushing data around with a broom.

Classes may be defined that are subclasses of another class. This causes objects of that class to inherit both the instance variables and actions of the superclass. Subclassing provides the capability of saying, "I want something just like that, except a little different." For example, you might want to have a **Pen** that recorded its movements. Since the current class **Pen** handles everything except the recording, subclassing **Pen** and creating a **RecordingPen** that provides the capability would make the most sense.

Since class **RecordingPen** is a subclass of **Pen**, **RecordingPen** objects will have both **Pen** and **RecordingPen** instance variables, and will respond to the union of all **Pen** and **RecordingPen** messages. In general, if the message is not defined in the class that the object belongs to, Neon will search up the superclass chain to find a class that can respond to the message, and runs the appropriate method. You may also add new messages and instance variables to distinguish your class from the superclass, and you can override superclass messages by providing them in your subclass.

To create the new **RecordingPen** class all you would need to do is add **startRecording:**, **stopRecording:**, and **playBack:** messages, and a new instance variable **penRecord**, which keeps the list of messages sent to the object since recording started. By overriding each message that the object wants to record, it can remember the message in **penRecord** and send the appropriate message to its superclass, **Pen**, to do the actual drawing.

Building a system from scratch based on classes and objects provides an extremely flexible, compact and modifiable system. Since the code that defines the system itself is accessible, extensions to the system may be made quickly and easily at even the lowest level.

(For an excellent discussion of classes, messages and object oriented programming, see Brad Cox's article "Message/Object Programming: An Evolutionary Change in Programming Technology" in the January 1984 issue of *IEEE Software.*)

Built-in Classes

Since Neon, like Smalltalk, was written in itself, using its own object and class concepts, the groundwork for building applications is already there and fully accessible to the programmer. The basic classes provided in Neon make it easy to support the Macintosh User Interface standard with minimal effort. In Pascal, a Mac program typically has several large case statements at the very highest

Periscope ... A Direct Hit!

"A great debugger ... If you've been frustrated by programs that don't behave the same for the debugger as when running alone, or that crash altogether, you should check out Periscope ..."

PC REPORT, Boston Computer Society

"It works, and works well!! In the first day of use I finished up two weeks of problems!! Really excellent!!!"

Peter Loats, Periscope User

Periscope's differences hit home! Its breakout switch gives you control, even when interrupts are disabled. Periscope's unique power helps you solve the difficult problems. Debug device-drivers, memory-resident, and non-DOS programs.

Great for straightening out confused C pointers! Using its breakpoint power, you can stop on reads and writes to ranges of memory. Stop on registers, words, and bytes, too.

It's tough. Periscope's 16K RAM board is write-protected. Runaway programs can't touch crucial debugger code! Examine the system, safely, at any time: Periscope saves the interrupt vectors and buffers it uses, then restores them when done. It uses ROM BIOS calls for functions except file access, so DOS can't clobber itself.

It gets you moving. It's commands are similar to Debug's, so you'll master it quickly. On-line help is there when you need it. You can define the windows you want; you can design easy-to-read memory displays. Periscope supports C, Assembler, BASIC, and Pascal. Comprehensive symbol support gives you a roadmap through memory tying back to your source.

It's risk-free. Periscope is backed up by a 30-day, money-back guarantee! The board is warranted for a year. Technical Support? You get the best there is ... direct from Brett Salter, Periscope's author!

It requires. An IBM PC, XT, AT, Compaq, or close compatible; PC-DOS, 64K RAM; a disk drive and an 80-column monitor. With two monitors, Periscope's screen is displayed on the monitor not in use. With one monitor, a keystroke switches screens.

It's power you can afford. At $295, you can afford Periscope's professional power. The system includes the memory board, breakout switch, debugger software, manual, and quick reference card.

Order now! Call toll-free (800) 722-7006 to place your order or to get more information. MasterCard/VISA accepted.

Get your programs up and running;

UP PERISCOPE!

Data Base Decisions/14 Bonnie Lane | Atlanta, GA 30328/(404) 256-3860

Circle no. 31 on reader service card.

©1985

4xFORTH™ for the Atari 520 ST

Multiuser, Multitasking 32 Bit Forth Based on the 83 Standard.

4xFORTH Level 1	$ 99.95
4xFORTH Level 2 with GEM	149.95
Forth Accelerator	75.00
4xFORTH Development System	500.00

The Dragon Group, Inc.
148 Poca Fork Road
Elkview, WV 25071
304/956-5517

Circle no. 146 on reader service card.

HISOFT

HIGH QUALITY SOFTWARE **CP/M**

HiSoft has been selling Z80 CP/M software in Britain and Europe for over 4 years. Now we'd like to introduce **you** to our range of programming languages:

HiSoft Devpac:	Z80 assembler/editor/debugger
HiSoft C:	Kernighan/Ritchie implementation
HiSoft Pascal:	fast, **standard** compiler

All at $69 inclusive each.

These programs are also available for other Z80 machines including Timex 2068.
Call or write for full technical details and press commentaries, or order from:

HISOFT
180 High St. North
Dunstable LU6 1AT
ENGLAND
01144 (582) 696421

Circle no. 48 on reader service card.

CROSS ASSEMBLERS
for the 8048 and 6502

Now you can use your PC with TASM65 to generate code for your old KIM/SYM/JOLT/AIM/OSI or homebrew 6502 system. Or, use TASM48 to develop code for 8048/8748/8035 based projects.

- For the MSDOS 2.X operating system
- Standard Mnemonics/address mode syntax
- Complex expression evaluation (17 operators)
- Object code output in Intel Hex. MOSTech Hex, or binary format
- Compatible with Aparat PROM blaster for inexpensive EPROM burning
- Source code available (in C)-table driven code easily modified for many other 8 bit microprocessors
- Absolute code generation only
- Unprotected disk
- 30 day money back guaranty

TASM — A Table Driven Assembler

TASM48 — An 8048 Assembler	$30.00 postpaid
TASM65 — A 6502 Assembler	$30.00 postpaid
Both, with source code	$75.00 postpaid

Washington State residents add 7.8% tax
Send check or money order to:

SPEECH TECHNOLOGY, INC.
16321 176th Ave. N.E.
Woodinville, WA 98072

Circle no. 136 on reader service card.

level of the main program to classify events received and to dispatch them to handlers. Neon helps to take care of this by encapsulating most of the standard event handling in class **Event**, menu selection and dispatching in class **Menu**, and standard window actions in class (you guessed it) **Window**. Through these classes, Neon shields the programmer from the standard necessities and deals with most of the housekeeping that is required in a Macintosh application behind the scenes.

There are many other classes written to simplify interaction with the User Interface Toolbox. Class **File** provides access to the Toolbox File I/O and Save/Open dialogs, and class **String** defines messages for printing, insertion, and pattern matching of text strings (a nice interface to Munger, the Toolbox string handler). In addition, there is a set of nine classes to support Quickdraw, including **Rect**, **Oval**, **Picture**, and **GrafPort**. All Toolbox routines are directly accessible.

Using Neon

Neon is started by double-clicking on **neon.com**. **Neon.com** is a prebuilt environment in which a basic set of classes, messages, and objects have been installed. As you work in this environment, you will be creating classes, objects and messages, as well as new Neon words. Once you have modified the default environment you may save it on the disk. In this way you can build your own custom Neon system.

When Neon starts up, it comes up with a single teletype window, where you can type Neon commands. Output from some of the utilities appears there also. The Neon window does not update when windows are moved around above it, which is a minor nuisance. There is no text editing allowed in the Neon window other than backspacing.

It is fairly easy to create new windows and menus while in Neon and to attach Neon words to each menu item. For example, sending the message **Example:** to a new window creates a Neon window that will interpret any text typed to it. There are also example messages in several other classes, including **Control**.

The standard **Neon.com** ystem, as it comes from Kriya, has five menus:

1) The **Apple** menu has the standard list of desk accessories, an "About Neon . . ." item, and an extra accessory called "Editor." This accessory provides the text-editing facilities necessary for editing class and object definitions within Neon.

2) The **File** menu consists of five items:
Load... loads a text source file into the current environment, defining new classes, objects, and messages. You would load **grdemo.load** to define the special classes for the graphics demonstration program.
Save saves your current environment onto the disk. As you modify classes and customize the system to meet your needs, you can save the complete system and return to it by simply opening the appropriate Neon document. This is often called "taking a snapshot."
Save As... allows you to save your current environment under a new name.
Print... prints a specified text file to the printer. The print command produces a formatted ("prettyprinted") listing of the source code in the file.
Quit closes all files and returns to the Finder. All changes made to the Neon environment are lost (unless they had been previously Saved.)

3) The **Edit** menu provides the standard **Undo**, **Cut**, **Paste**, **Copy**, and **Clear** items, as well as a special **Editor** item to start up the editor desk accessory.

4) The **Utilities** menu has six items:
List Words lists all words currently defined in the dictionary.
List Objects lists all objects of a given type that currently exist in the environment.
List Classes lists all defined classes in the current environment, indenting to show subclass relationships.
Examine Memory is a feature that allows you to examine and search the Macintosh memory.

Grep... searches all text files on the disk for a given string. The results of the search (file name, line number, and text line) are printed in the Neon window.
Install lets you change the sizes of the stack, dictionary, and heap. The stack and dictionary are adjustable, while the heap gets what is left.

5) Finally, the **Neon** menu provides five items:
Echo to Printer echoes all text that appears in the Neon window to the printer, including text that is typed by the programmer. This is especially useful in that it allows the programmer to keep a record of the input and output of a program being debugged.
Echo During Load displays the text from a source file being loaded in the Neon window as it is being compiled. This is useful in helping the programmer debug a new module or class definition.
Show Free Space displays the amount of memory available in the dictionary and the heap in the Neon window.
The last two entries in the Neon menu deal with modules. Modules are separately compilable, relocatable code segments that are loaded and purged from memory dynamically. This allows large applications to run without having to be completely resident in memory. These modules act similarly to the Macintosh code segments in that they are dynamically loaded into memory and purged when not in use. Many of the menu items in Neon are defined by modules, including **Grep** and **Examine**.
Show Module Status lists all purgeable modules currently defined in the dictionary and shows whether they are loaded by printing their memory locations (or 0 if not loaded).
Purge Modules purges all modules from the heap, freeing up space for a memory-intensive operation.

Because the Neon window provides no editing capability, the system comes with an editor to edit class definitions within Neon. This editor is an enhanced MockWrite desk accessory, from CE Software. I found it easy and convenient to use, though I would prefer a built-in editor in which I could select, edit, and execute Neon statements directly. Once a class definition is created, it may be loaded into the system by either typing // followed by the filename or by selecting the **Load** item in the file menu. Either action begins compilation of the file. After the file has been completely compiled and defined in the system, you may create new objects, send messages to previously-created objects, and so on.

After a few hours with Neon, I felt quite at home creating objects, sending messages, and building graphical tools. The fact that Neon is an interactive programming environment is a great advantage in building applications for the Macintosh because the turn-around time is much shorter and experimentation is easier. Neon also provides a facility for "sealing off" an application, creating a double-clicking program without access to the Neon environment.

Forth Improvements
Besides support for objects and classes, Neon has other improvements to standard Forth. The main objection to Forth is usually the need for strange and unintelligible stack manipulation in function definitions; Neon avoids this through local variables and named parameters. Consider the example in the Neon tutorial for calculating **a2 + b2:**

```
:formula1 { a b -- solution }
a a * → a
b b *
a + . CR ;
```

The input parameters are named between the braces, and are pulled off the stack and stored in the local variables **a** and **b**. The arrow (→) stores the value on the stack into the named parameter. Local variables may also be defined within the braces by preceding the variable list with a backslash. Again, from the Neon tutorial, consider the formula **(a + b-3c)/(b + 3c):**

```
:formula2{ a b c \ num den -- result }
```

Does your **ISAM** run on **IBM, APPLE, DEC** and **AT&T** computers? **c-tree** does, and you only **BUY IT ONCE!**

c-tree™
BY FAIRCOM
2606 Johnson Drive
Columbia MO 65203

The company that introduced micros to B+ Trees in 1979 and created ACCESS MANAGER™ for Digital Research, now redefines the market for high performance, B+ Tree based file handlers. With c-tree™ you get:

- complete C source code written to K&R standards of portability
- high level, multi-key ISAM routines and low level B+ Tree functions
- routines that work with single-user and network systems
- no royalties on application programs

$395 COMPLETE

Specify format:
5¼" PC-DOS 3½" Mac
8" CP/M® 8" RT-11

for VISA, MC or COD orders, call
1-314-445-6833

Access Manager and CP/M are trademarks of Digital Research, Inc. Apple is a trademark of Apple Computer, Inc. c-tree and the circular disc logo are trademarks of FairCom IBM is a trademark of International Business Machines Corporation DEC is a trademark of Digital Equipment Corporation ©1984 FairCom

Circle **no. 37** on reader service card.

PRIME FEATURES

- Execute DOS level commands in HS/FORTH, or execute DOS and BIOS functions directly.
- Execute other programs under HS/FORTH supervision. (editors debuggers file managers etc)
- Use our editor or your own.
- Save environment any time as .COM or .EXE file.
- Eliminate headers, reclaim space without recompiling.
- Trace and decompile.
- Deferred definition, execution vectors, case, interrupt handlers.

HS/FORTH

- Full 8087 high level support. Full range transcendentals (tan sin cos arctan logs exponentials)
- Data type conversion and I/O parse/format to 18 digits plus exponent.
- Complete Assembler for 8088, 80186, and 8087.
- String functions - (LEFT RIGHT MID LOC COMP XCHG JOIN)
- Graphics & Music
- Includes Forth-79 and Forth-83
- File and/or Screen interfaces
- Segment Management
- Full megabyte - programs or data
- Fully Optimized & Tested for: IBM-PC XT AT and JR COMPAQ and TANDY 1000 & 2000 (Runs on all true MSDOS compatibles!)
- Compare

BYTE Sieve Benchmark jan 83
HS/FORTH 47 sec BASIC 2000 sec
with AUTO-OPT 9 sec Assembler 5 sec
other Forths (mostly 64k) 55-140 sec

FASTEST FORTH SYSTEM AVAILABLE.
TWICE AS FAST AS OTHER FULL MEGABYTE FORTHS!
(TEN TIMES FASTER WHEN USING AUTO-OPT!)

HS/FORTH, complete system only: $270.

Visa Mastercard

HARVARD SOFTWORKS

P.O. BOX 69
SPRINGBORO, OH 45066
(513) 748-0390

Circle **no. 44** on reader service card.

```
a b + 3 c * → num
3 c * b + → den
num den / ;
```

The numerator and denominator are calculated separately, stored in local variables, and then the formula is calculated. With local variables and named parameters, Neon programs are downright readable.

Summary

Neon is nicely designed. It is clear that the developers had an excellent understanding of what object-oriented programming is supposed to be. With the basic classes, examples, and tutorial, you could be writing interesting programs that support the Macintosh User Interface Standard in a matter of weeks. This compares very favorably with the months that most people take to build the support routines that a real application requires. Kriya is also planning to support a clearinghouse service for useful, public domain classes written in Neon for use by software developers.

Neon is also a good programming language for occasional use, though it has so many features and is so flexible that it could take a little while to learn what is provided in the system. The documentation is thorough and well-written, and the tutorial is interesting and useful.

As a programming environment, Neon is very well suited to the Macintosh. The built-in classes are well designed and useful, and provide a much improved interface to the Macintosh Toolbox routines. It is flexible, powerful, fast and a lot of fun. It is also the only implementation of an object-oriented system for the Macintosh so far. Because of the support for classes and objects, organization of large programs would be quite manageable, while with standard Forth working on a large system with thousands of words might become confusing. The only disadvantage that Neon has, in my opinion, is the backwards Forth syntax. If somebody comes up with a new infix parser-scanner to allow Smalltalk-like expressions, Neon could be a real winner in the programming language sweepstakes.

MacFORTH Level 1, Version 1.2; Level 2, Version 2.1; Level 3, Version 3.0

**Company: Creative Solutions, Inc.
4701 Randolph Rd, Ste. 12
Rockville, MD 20852
(301) 984-0262
Computer: Apple Macintosh (128K)
Price: Level 1—$149, Level 2—$249 (includes Level 1), Level 3—$499 (includes Levels 1 and 2)**
(Circle Reader Service No. 103)
Reviewed by Alan Clute

The Macintosh is a giant toolkit. An ideal language to take advantage of this toolkit is Forth. MacFORTH, from Creative Solutions, Inc., is an excellent Forth for the Macintosh. MacFORTH delivers the richness and power of the Macintosh without getting overly complex. The package, while being simple to use, still leaves the programmer enough breathing room to develop programs that are powerful and make good use of the Mac's capabilities.

MacFORTH is a production language. Current products written in MacFORTH include PeachTree's *Back to Basics Accounting*, Brainworks' *ChipWits*, and Ideaform's *MacLabeler*. An interesting new product is CSI's *StudioMac*, a music program that includes tutorials, source code, MIDI interface capability, and complete plans to build an inexpensive MIDI interface for the Macintosh.

Contents of Package

Table 1 (see page 101) shows the main contents of each level of MacFORTH. The three levels are nested, and upgrades are available for the difference in list price. Because Level 2 includes more and is in some ways easier to work with than Level 1, it is probably the best package to get first, especially at mail-order prices. For the record, this review concerns itself with Versions 1.2, 2.1, kernel 2.3, and, briefly, with Level 3 version 3.0.

MacFORTH has been around a fairly long time, as Macintosh software goes. Level 1 was first released in

April of 1984, with the first version of Level 2 coming out in July of that year. MacFORTH was the first serious native language for the Macintosh. It makes good use of a 128K Macintosh and can be used with only one disk drive (if you are so inclined).

MacFORTH is not copy protected, so it can easily be backed up or installed on a hard disk. The accompanying license agreement says in no uncertain terms that you should not give or sell copies to others, but CSI is to be congratulated for not otherwise interfering with your use of the program.

Forth Implementation

MacFORTH is derived from the Forth-79 standard, optimized for the Macintosh environment. It is a 32-bit Forth, which is more suitable for the large address space of the Mac than 16-bit implementations. Considered just as Forth, MacFORTH is a good one. Converting programs from another standard version of Forth should be rather straightforward.

The internal implementation is direct threaded with a token table to keep all word references to 16 bits. Thus, program code requires about the same amount of dictionary space as in a 16-bit version of Forth. Variables occupy 32 bits. Fetch and store operators come in several flavors to support 8, 16, or 32 bit values.

MacFORTH uses traditional Forth blocks for source code, but these blocks are within block files. MacFORTH thus lives very comfortably within the Macintosh environment.

Programs can do disk I/O using block files, or can choose to use data files with text, fixed-length record, or byte-addressable virtual formats. The entire storage-management vocabulary includes about 90 words.

Documentation

MacFORTH is extremely well documented. Most of the documentation covers the Macintosh interface, but then, most of the Forth is quite standard. The level 1 manual is about 160 pages, plus glossary and appendices. Level 2 runs over 200 pages, plus glossary, and Level 3 adds 200 more pages, which show you step by step how to turn a running program into a production disk.

The manuals make clear presentations. The glossaries are well organized, well indexed, and well written. The only thing obviously missing is a reference in each glossary entry to the page where the word is discussed in context.

MacFORTH pretty much assumes you know Forth, although it comes with a Forth tutorial and a rich set of examples that would prove useful even to a beginner. As is often advised, the beginner really should arm him/herself with some instructional book like *Starting FORTH* or *Thinking FORTH* by Leo Brodie (Prentice-Hall, 1981 and 1984) instead of relying on a language manual.' In addition, a new book should be out by the time you read this, *MacFORTH: Programming for the Rest of Us* by Dave Colburn, one of the authors of MacFORTH (also available from CSI).

The Going Forth tutorial, a split-screen affair with paged text on one side and the live MacFORTH system on the other, provides a hands-on introduction to MacFORTH. Source is on disk for those who want to expand the tutorial for others.

You get plenty of source code with these packages, both "real-world" stuff, like the block editor, and "tutorial" material like the Note Card

Level	Feature	Comments
1	• Going FORTH tutorial	Disk based, interactive.
	• Block-oriented editor	Similar to MacWrite.
	• Macintosh interface	Menus, windows, mouse, sound, event handler.
	• File system	Text, record, virtual, and block files. MacWrite text files.
	• Printer/Serial interface	Window or screen output, full serial I/O.
	• Programming aids	Trace, debug, error traps, direct calls to toolbox routines.
	• Source code	Forth extensions, editor, many examples, terminal program, download conversion, standard file-open dialogue.
2	• Level 1 included	
	• System tools	Heap management, 68000 assembler, snapshot (vocabulary save).
	• Advanced graphics	Polygons, regions, pictures, rectangle arithmetic and scrolling.
	• Text editing	TextRecords and TextFields.
	• Controls	Buttons, boxes, and scroll bars.
	• Clipboard I/O	Read and write from/to clipboard
	• Floating Point	IEEE-754 standard
	• Source code	Graphics demo, note card data management demo (fully annotated in manual), many examples, source for above extensions.
3	• Level 2 included	
	• Application workshop	Turnkey production tools and tutorial. Supports 512K Mac, overlays on 128K Mac.
	• Resource workshop	Macintosh resource customization and tutorial.
	• Kernel workshop	Forth kernel customization and tutorial.

Table 1:
Contents of MacFORTH Levels

example. This last is part of Level 2 and includes 23 screens of source and 20 pages of detailed manual text. In all, Level 1 includes about 120 screens of source, and level 2 about 140 screens.

Forth and the Macintosh
Forth has been described as a language for the development of problem-oriented languages. MacFORTH is a Macintosh-oriented language. The "problems" of the Macintosh—how to use windows, pull-down menus, graphics, and other features—have been smoothly addressed by MacFORTH.

The MacFORTH vocabulary is huge, over 900 words in Level 1 with another 250 or so added in Level 2. Most of these are the "application words" that give you direct control over the Macintosh. The Macintosh and its fabled toolbox are made directly available, letting you do things "the Macintosh way" with relative ease. Don't be intimidated by the large vocabulary: flexibility comes from lots of little words, rather than a few large ones, and the Macintosh has many of these small units. The window-handling package alone has over 70 words associated with it.

```
        ( create window )
15 CONSTANT testWindow
NEW.WINDOW testWindow

        ( define title )
   " Example Window"                  testWindow W.TITLE
        ( define rectangle )
   50 50 100 300                      testWindow W.BOUNDS
        ( add boxes )
   CLOST.BOX SIZE.BOX +               testWindow W.ATTRIBUTES
        ( add window to the screen )
   testWindow ADD.WINDOW
```

Figure 1
Code to Create Window

```
        ( create menu, define title )
11 CONSTANT testMenu
0 " Example" testMenu NEW.MENU
        ( define item text )
" First;Second;Third(;" testMenu APPEND.ITEMS
        ( add menu to the screen )
DRAW.MENU.BAR
```

Figure 2
Menu Definition

```
        ( define menu handler )
testMenu MENU.SELECTION:
        ( CASE statement for items )
   CASE 1 OF FirstProgram ENDOF
          2 OF SecondProgram ENDOF
          3 OF ThirdProgram ENDOF
   ENDCASE
        ( turn off title hilite )
   0 HILITE.MENU ;
```

Figure 3
Selection of Menu Item

Examples
Straightforward use of windows and pull-down menus, for example, turns out to be simplicity itself. The code in Figure 1 (at left) creates a window with a close box and a size box. Figure 2 (at left) shows an example menu definition, with 3 items. The first two are "active" and the third is "inactive" (grayed out and incapable of being selected). The trailing **(** in **Third(** causes that item to be inactive. Type style (bold, underline, etc.) and command key equivalents are easily assigned using other character suffixes. Selection of a menu item is easily linked to the proper actions using a defining word (Figure 3, at left). Fancier uses of scrollbars, control buttons, and the like are implemented with the clarity and simplicity so essential to good Forth style.

The Level 2 manual includes a reference section giving the Pascal name, MacFORTH name, and stack usage for the over 150 toolbox traps directly callable by the user. For advanced applications, MacFORTH provides simple low-level words to access any toolbox routine.

Support
CSI maintains a technical hotline for owners of Level 1 and Level 2, and publishes a quarterly newsletter. The hotline is open during business hours and is a normal toll call.

An important part of the MacFORTH "safety net" is the Special Interest Group (SIG) on CompuServe. This is currently open to registered owners of MacFORTH only, and according to CSI about 10% of MacFORTH owners participate. As with most such SIGs, this one offers data libraries with public domain software written in MacFORTH. One data library is devoted to notices and patches for known bugs. A terminal program (with source code) is included on the Level 1 disk.

User Interface
You would expect a first-class Macintosh toolkit implementation to make very good use of the Macintosh, and, for the most part, MacFORTH does just that. Windows and menus for the MacFORTH environment are stan-

dard Macintosh. The text editor included in Level 1 is a subset of MacWrite, with source on disk if you want to study it or add features.

MacFORTH is so Mac oriented that rough spots in the user interface come as a bit of a shock. For example, the Forth text interpreter is plain vanilla Forth and doesn't seem to know about mousing around. Keystrokes just sort of lie there on the screen, and to edit a mistake you have to back up and retype. Gasp! Fortunately the CompuServe SIG comes to the rescue here with a handy little utility that stores the last ten commands, in editable form, in a pull-down menu. Nothing like a user's group to rescue one from primitive conditions.

Another minor annoyance is that text in the MacFORTH window is not restored after being overlaid by the editor window. When you exit the editor you are left with a blank rectangle chopped out of your dialogue with MacFORTH. This conserves memory, but is tacky. Those of you with 512K Macs will want to add screen save/restore to the editor with a few lines of code.

Developer Package, Level 3

I was only able to look at the manual for Level 3. This package is for developers who want to produce turnkey products for distribution. CSI used to charge a per-copy royalty, but the new Level 3 package replaces that policy and includes a royalty-free license agreement.

The Level 3 manual provides some excellent tutorial material and seems to be as concerned with helping you turn out a good stand-alone product as the Level 1 is with helping you make good use of the Macintosh.

The support situation is a bit different for Level 3. CSI is understandably wary of spending a lot of time on the phone with developers, so Level 3 owners are expected to use the CompuServe SIG instead of the CSI hotline when they need help. In practice, this seems to work out fine, to judge from the message traffic on the SIG (which includes heavy participation by CSI staff). If you are planning to use MacFORTH to develop a turnkey product or program for distribution, you might buy Level 2, use the hotline and the SIG while learning MacFORTH and during program development, and then upgrade to Level 3 when you are in the final stages.

Conclusion

It is clear that the authors of MacFORTH want you to get your hands on the Macintosh and use it well. The manuals, samples, and examples all support this "Power to the People" approach. Most of the manual is about how to use the Macintosh with the tools provided by MacFORTH. The result is that, for the most part, you don't have to spend hours or days poring over *Inside Macintosh* to get the Macintosh to do something, but can instead learn by example and hands-on experimentation. Because it is so easy to try things in Forth, using it on the Mac should lead to even greater reduction of program development time than would using Forth on a more limited machine.

DDJ

How much time can you buy for 49^{95}?

You know what your time is worth, but **MATIS**™ users know that they save long hours everyday in developing an attractive and efficient user interface for their programs.

MATIS is a DOS extension consisting of assembler routines with the interfaces to BASIC, C, PASCAL, and ASSEMBLER.

If you think you spend too much time:

☐ Creating Windows ☐ Defining large Virtual Screens ☐ Formatting Input Fields ☐ Scrolling in four directions ☐ Setting independent Video Attributes ☐ Controlling Keyboard input at run-time ☐ Drawing lines and boxes ☐ Printing and maintaining your screens.

Buy yourself a well deserved rest!
With a copy of MATIS for only $49.95

NEW! **MATIS/T**™ for Turbo-Pascal only **$29.95**! Same features as the original MATIS. An indispensable add-on at a dynamite price.

ORDER BY MAIL*—WRITE OR CALL FOR COMPLETE DESCRIPTION
No License Fee

Softway, Inc.

500 Sutter Street • Suite 222-K • San Francisco, CA 94102
(415) 397-4666

Credit Card Orders Only, Call 7 days a week, 24 hours a day (800) 227-2400 Ext. 989

*ADD $5.00 FOR SHPG CA RES. ADD SALES TAX

Circle **no. 92** on reader service card.

ANNOUNCING! DR. DOBB'S COMPLETE TOOLBOX OF C

Dr. Dobb's Journal, the most respected source of technical software information available, brings you this collection of powerful programming tools.

Available in October from M&T Publishing and Brady Communications—

Dr. Dobb's Toolbook for C

A comprehensive library of valuable C code

Many of Dr. Dobb's most popular articles on C from sold-out issues are updated and reprinted in this unique reference, along with new C programming tools. **THE TOOLBOOK** contains a complete C compiler, an assembler, text processing programs, and more! Dr. Dobb's Journal offers **THE TOOLBOOK** in a special hardbound edition for only $29.95. You'll find:

- James Hendrix's famous Small C Compiler v. 2 and A New Library for Small C (also available on disk)
- Ron Cain's original A Small-C Compiler for the 8080's
- Never before published: Hendrix's new *Small-Mac: An Assembler for Small C* and *Small Tools: Programs for Text Processing* (both available on disk)
- Plus many useful programming tools in C

Also from M&T Publishing and Brady Communications—

The Small-C Handbook

The Small-C Handbook by James Hendrix is a valuable source of information about the Small-C compiler. In addition to descriptions of the language and the compiler, **The Handbook** also contains the entire source listings of the compiler and its library arithmetic and logical routines.

A perfect companion to the Hendrix Small-C compiler offered by Dr. Dobb's on disk, **The Handbook** even tells how to use the compiler to generate new versions of itself. $17.95

Dr. Dobb's C Tools On Disk

To complement **The Toolbook** Dr. Dobb's also offers the following programs on disk for only $19.95 each. Full source code is included and, except where indicated, both CP/M and MS or PC DOS versions are available.

Small-C Compiler

Jim Hendrix's **Small-C Compiler** is the most popular piece of software published in Dr. Dobb's 10-year history. Like a home study course in compiler design, the **Small-C Compiler** and **The Small-C Handbook** provide all you need to learn how compilers are constructed, as well as teaching the C language at its most fundamental level. **The Small-C Handbook** provides documentation for both versions ; however, an addendum is recommended in addition to **The Handbook** for MS or PC DOS-specific documentation. The addendum is available for $4.95.

Small Tools: Programs for Text Processing

This package consists of programs designed to perform specific functions on text files, including: editing; formatting; sorting; merging; listing; printing; searching; changing; transliterating; copying and concatenating; encrypting and decrypting; replacing spaces with tabs and tabs with spaces; counting characters, words, or lines; and selecting printer fonts. Documentation available for $9.95.

Small-Mac: An Assembler for Small-C

Small-Mac is a macro assembler designed to stress simplicity, portability, adaptability, and educational value. The package features simplified macro facility, C-language expression operators, descriptive error messages, object file visibility, and an externally defined machine instruction table. Included programs are: macro assembler, linkage editor, load-and-go loader, library manager, CPU configuration utility, and dump relocatable files. This program is available for CP/M systems only. Documentation available for $9.95.

To order, send this order form with your payment to: **Dr. Dobb's Journal, 2464 Embarcadero Way, Palo Alto, CA 94303**

	CP/M	MS or PC DOS			QTY.		TOTAL
BOOKS							
Dr. Dobb's Toolbook`			$29.95	X		=	
Small-C Handbook			$17.95	X		=	
DISKS (Check Format)							
Small-C Compiler			$19.95	X		=	
Small Tools Text Processor			$19.95	X		=	
Small Mac Assembler (for CP/M only)		▨	$19.95	X		=	
MANUALS							
Small-C Compiler MSPC-DOS specific addendum to The Small-C Handbook (NOTE: The Small-C Handbook provides full documentation for the CP/M version)			$4.95	X		=	
Small Tools			$9.95	X		=	
Small Mac			$9.95	X		=	

Sub Total $

California residents add applicable sales tax — Tax

Add $1.75 per item for shipping in U.S., $4.25 per item outside U.S. — Shipping

TOTAL $

For CP/M system disks only, please specify one of the following formats:

- ☐ Kaypro
- ☐ Apple
- ☐ Zenith Z-00 DS/DD
- ☐ Osborne
- ☐ 8″ SS/SD
- ☐ Inquire about other formats

PAYMENT MUST ACCOMPANY YOUR ORDER

☐ Check enclosed

☐ Please charge my ☐ VISA ☐ M/C ☐ Amer. Exp.

Card #____________________

Exp. Date____________________

Signature____________________

Name____________________

Address____________________
(please use street address)

City____________________

State____________Zip____________

Allow 6–12 Weeks for delivery 3108

Last month (September 1985, #107 page 94) John Bass described how to put together your own hard-disk interface for the Macintosh for less than $500. The following listing, np_dvr.c, is the driver routine.

Mac Toolbox Listing

```
/*
 * np_dvr.c version 1.0, July 20, 1985
 * MacSCSI non-partitioned driver routine.
 * This version was based on the ramdisk.asm example found in
 * Aztec C 1.06 and was developed on the Mac under Aztec C.
 *
 * The changes are Copyright 1985 by John L. Bass, DMS Design
 * PO Box 1456, Cupertino, CA 95014
 * Right to use, copy, and modify this code is granted for
 * personal non-commercial use, provided that this copyright
 * disclosure remains on ALL copies. Any other use, reproduction,
 * or distribution requires the written consent of the author.
 *
 * Sources are available on diskette from Fastime, PO Box 12508
 * San Luis Obispo, Ca 93406 -- (805) 546-9141. Write for ordering
 * information on this and other Mac products.
 */

#asm
;:ts=8
;
DRVNUM  equ     5                       ;change this if conflicts with others
;
Start
        dc.w    $4f00                   ;locked, read, write, ctrl, status
        dc.w    0                       ;no delay
        dc.w    0                       ;no events
        dc.w    0                       ;no menu
        dc.w    open-Start              ;open routine
        dc.w    rdwrt-Start             ;prime routine
        dc.w    control-Start           ;control routine
        dc.w    status-Start            ;status routine
        dc.w    tst-Start               ;close routine
        dc.b    8
        dc.b    ".MacSCSI"              ;name of driver
        ds      0                       ;for alignment
;
MAXMAP          equ     640
;
drBlLen         equ     16
drNmAlBlks      equ     18
drAlBlkSiz      equ     20
drClpSiz        equ     24
drAlBlSt        equ     28
drFreeBks       equ     34
;
template
        dc.w    $d2d7                   ;signature word
        dc.l    0                       ;init time
        dc.l    0                       ;backup time
        dc.w    0                       ;attributes
        dc.w    0                       ;# of files
        dc.w    4                       ;first dir block
        dc.w    0                       ;# of dir blocks
        dc.w    0                       ;# of allocation blocks total
        dc.l    0                       ;bytes/allocation block
        dc.l    0               ;bytes/allocation call
        dc.w    4               ;first data block
        dc.l    1               ;next free file #
        dc.w    0               ;# of unused allocation blocks
        dc.b    7               ;length of name
        dc.b    "MacSCSI"               ;name of drive
        ds      0               ;for alignment
;
stabuf
        dc.w    0               ;current track
        dc.b    0               ;write protect in bit 7
        dc.b    1               ;disk in place
        dc.b    1               ;drive installed
        dc.b    0               ;sidedness of drive
        dc.l    0               ;next queue entry
        dc.w    0               ;unused
        dc.w    DRVNUM          ;drive number
drvref  dc.w    0               ;driver ref num
        dc.w    0               ;file system ID
        dc.b    0               ;sidedness of disk
        dc.b    0               ;needs flush flag
        dc.w    0               ;error count
;
diskbase_
        dc.l    0               ;base of disk memory
;
openflg
        dc.w    0               ;once only open flag
;
open
        lea     openflg,a0      ;get once only flag
        tst.w   (a0)            ;is it open already?
```

(Continued on next page)

SOFTWARE DEVELOPERS

Save thousands of dollars! Save hundreds of hours!

by using our assembly language sub-systems

B-TREE SUB-ROUTINES

FABS

Internationally known and used in many best selling application programs . . . Rapid access and maintenance of large files with fixed-length records . . . Versions available for CP/M-80, CP/M-86, MP/MII, MS DOS, PC DOS, Microsoft BASIC(s), COBOL, FORTRAN, PASCAL, PL/I, CBASIC, CB80, CB86, CBASIC, CBASIC 86, LATTICE C.

RETAILS FOR $150 DEALER/OEM PRICES AVAILABLE

FABS PLUS

Expanded version of our FABS products . . . Up to millions of records DEP on Key Size . . . Extremely fast on unlimited number of keys . . . Re-indexing program included . . . Can be used on files as large as your system can hold.

RETAILS FOR $195 DEALER/OEM PRICES AVAILABLE

SORT/MERGE SUB-ROUTINES

AUTOSORT

Optimized for very large files; stand-alone or callable sub-routine, extremely fast . . . Versions available for CP/M 80, CP/M 86, MP/MII, MS DOS, PC DOS running Microsoft BASIC(s), FORTRAN, PASCAL, CBASIC, CBASIC 86, CB 80, CB 86, LATTICE C.

RETAILS FOR $150 DEALER/OEM PRICES AVAILABLE

DATA, SCREEN, REPORT MANAGER

DB-FABS

A highly capable DATA BASE package designed to perform for everyone from the novice user in the Stand-Alone mode to the professional programmer in the Run-Time mode . . . Creates files, forms, reports, handles screening . . . B-Tree Indexing, high speed sorting capabilities . . . Run-Time mode use with BASIC INTERPRETER/COMPILER . . . For MS DOS, PC DOS on IBM PC/XT, DEC Rainbow, Victor 9000, Sanyo, Fujitsu, etc.

RETAILS FOR $295 DEALER/OEM PRICES AVAILABLE

For more detailed information concerning any of our products, please contact us:

COMPUTER CONTROL SYSTEMS, INC

Route 3, Box 168, Lake City, FL 32055 (904) 752-0912

Circle **no. 18** on reader service card.

Mac Toolbox Listing

(Listing continued, text begins on page 94 of September Issue)

```
        bne      skip               ;yes -- just exit
        st       (a0)               ;set once only flag

        move.l   #14,d0             ;size of drive queue entry
        dc.w     $a51e              ;NewPtr - system heap
        clr.w    10(a0)             ;local file system
        move.l   #DRVNUM,d0         ;drive number 5
        swap     d0
        move.w   24(a1),d0          ;driver reference number
        lea      drvref,a2          ;get address of driver reference number
        move.w   d0,(a2)            ;set up status buffer
        dc.w     $a04e              ;AddDriver

        move.l   (a1),-(sp)         ;push handle to resource
        dc.w     $a992              ;DetachResource

        link     a6,#-50            ;allocat the block on the stack
        move.l   sp,a0              ;setup as arg to mountvol trap
        move.w   #DRVNUM,22(a0)     ;set our drive number in block
        dc.w     $a00f              ;ask for it to be mounted
        unlk     a6                 ;clear block from stack

skip
        move.l   #0,d0              ;set non-error return
        rts                         ;exit back to caller
;
rdwrt
        movem.l d0-d7/a0-a6,-(sp)          ;save regs
```

Time and Money.

We've just done something we know you'll like. We've made the SemiDisk far more affordable than ever before. With price cuts over 25% for most of our product line. Even our new 2 megabyte units are included.

It's Expandable

SemiDisk Systems builds fast disk emulators for more microcomputers than anyone else. S-100, IBM-PC, Epson QX-10, TRS-80 Models II, 12, and 16. You can start with as little as 512K bytes, and later upgrade to 2 megabytes per board...at your own pace, as your needs expand. Up to 8 megabytes per computer, using only four bus slots, max! Software drivers are available for CP/M 80, MS-DOS, ZDOS, TurboDOS, VALDOCS 2, and Cromix. SemiDisk turns good computers into **great** computers.

Battery Backup, Too

At 0.7 amps per 2 megabytes, SemiDisk consumes far less power than the competition. And you don't have to worry if the lights go out. The battery backup option gives you 5-10 hours of data protection during a blackout. Nobody else has this important feature. Why risk valuable data?

The Best News

	512K	1Mbyte	2Mbyte
SemiDisk I, S-100	$695	$1395	
SemiDisk II, S-100	$995		$1995
IBM PC, XT, AT	$595		$1795
QX-10	$595		$1795
TRS-80 II, 12, 16	$695		$1795
Battery Backup Unit	$150	$150	$150

**Someday you'll get a SemiDisk.
Until then, you'll just have to....wait.**

SEMIDISK

SemiDisk Systems, Inc., P.O. Box GG, Beaverton, Oregon 97075 503-642-3100

MasterCard VISA

Call 503-646-5510 for CBBS/NW, 503-775-4838 for CBBS/PCS, and 503-649-8327 for CBBS/Aloha, all SemiDisk equipped computer bulletin boards (300/1200 baud) SemiDisk, SemiSpool trademarks of SemiDisk Systems.

Circle **no. 85** on reader service card.

```
        jsr     ScsiRdWr_                  ;Call the C stuff
        movem.l (sp)+,dØ-d7/aØ-a6          ;restore regs
        bra     tst                        ;exit via IOdone

control
        cmp.w   #1,26(aØ)        ;is it KillIO
        bne.s   tst              ;no, exit
        move.l  #Ø,dØ
        rts                      ;just return
;
status
        cmp.w   #8,26(aØ)        ;is it status request??
        bne.s   tst
        movem.l aØ/al,-(sp)      ;save regs
        lea     28(aØ),al        ;get dest address
        lea     stabuf,aØ
        move.l  #22,dØ           ;move 22 bytes
        dc.w    $aØ2e            ;BlockMove
        movem.l (sp)+,aØ/al      ;restore regs
;
tst
        move.l  #Ø,dØ                      ;okay return
        movem.l d4-d7/a4-a6,-(sp)          ;save regs going into IOdone
        move.l  $8fc,aØ                    ;get IOdone address
        jsr     (aØ)                       ;call IOdone
        movem.l (sp)+,d4-d7/a4-a6          ;restore them after IOdone
        rts                                ;and jump to it

        public  _Uend_,_Dorg_,_Cend_

save_
        lea     Start+(_Uend_-_Dorg_)+(_Cend_-Start),a4 ; setup baseadr
```

(Continued on next page)

The Complete and Comprehensive Intelliware™ Laboratory for Expert System Concepts on the IBM Personal Computer.

Experteach Includes:
- Comprehensive Introduction to Expert System Concepts.
- On-line Tutorial Describing the Operation of Expert Systems.
- Lisp Based Expert System Tools with Source Code.
- Prolog Based Expert System Tools with Source Code.
- dBASE II™ Based Expert System Tools with Source Code.
- Pascal Based Expert System Tools with Source Code.
- Complete Lisp Interpreter for the IBM PC®.
- Complete Prolog Interpreter for the IBM PC.
- Comprehensive Case Studies of Several Major Expert Systems.
- Comprehensive bibliography on Expert Systems.

Experteach is a comprehensive guide to Expert System technology consisting of a uniquely integrated collection of Expert System tutorials, case studies, on-line teaching programs, Expert System building tools with source code and Artificial Intelligence languages. Experteach is based on extensive experience in teaching Expert System concepts in association with IEEE, ACM and the Continuing Education Institute.

Experteach introduces you to Expert System technology by allowing you to build Expert Systems and to experiment with a variety of Artificial Intelligence tools and languages on the IBM PC.

Experteach includes eight rule-based Expert System shells with source code implemented in Lisp, Prolog, dBASE II and Pascal. Each language has been used to implement both a forward chaining and a backward chaining Expert System shell with a built-in rule editor, inexact inference and how & why explanation facilities.

Experteach includes a comprehensive Lisp interpreter and a complete Prolog interpreter with DEC-10 Prolog syntax. Experteach requires only 256K of memory.

Intelliware, Inc., 4676 Admiralty Way Suite 401 Marina del Rey, CA 90291 (213) 827-1334

☐ Introduction to Expert System Concepts, On-line Tutorial, Case Studies, Bibliography and Pascal Based Tools $99.00.

☐ Lisp, Prolog or dBASE II Based Tools $99.00 each.

☐ Complete Experteach System with Lisp and Prolog Interpreters. $475.00 Check, Money order, Visa or Mastercard. $9.00 for postage and handling. California, 6.5% tax.

Intelliware, Inc.

Experteach and Intelliware are trademards of Intelliware, Inc. IBM PC is a registered trademark of IBM. dBASE II IS A TRADEMARK OF Ashton-Tate

Circle **no. 73** on reader service card.

A Professional Quality Z80/8080/8085 Disassembler

WHEN YOU NEED SOURCE FOR YOUR CODE

you need REVAS 3

REVAS interactively helps you:

Analyse your software for modification
disassemble files as large as 64K
Assign Real labels in the disassembly
Insert COMMENTS in the disassembly
Generate a Cross Reference (XREF) listing

A 60 page manual shows how the powerful **REVAS** command set gives you instant control over I/O to files, printer, or console; how to do a disassembly; and even how the disassembler works! You get on line help, your choice of assembler mnemonics, control of data interpretation, and calculation in any number base!

REVAS runs in Z80 CPM computers; is available on 8" SSSD (standard), RAINBOW, and other (ask) formats

Price: $90.00 (plus applicable tax), Manual only: $15.00

REVASCO
6032 Chariton Ave., Los Angeles, CA 90056
Voice: (213)649-3575 Modem: (213)670-9465

Circle **no. 80** on reader service card.

```
        move.l  a0,Pbp_
        move.l  a1,Dp_                          ;save arguments
        rts

restore_
        move.l  Dp_,a1                          ; pass DCEptr
        rts
#endasm

#include        <memory.h>
#include        <resource.h>
#include        <desk.h>
#include        <pb.h>


/*
 * varibles used by asm routines
 * Dp and Pbp are device driver arguments handled by save/restore
 * ptr and dvrarg needed by AddDriver call
 */

DCEPtr          Dp;
ParmBlkPtr      Pbp;

/*
 * Local Stuff
 */

int     DvrState;
```

TURBO EDITASM

Introducing the first co-resident editor assembler for the IBM PC family. ***TURBO EDITASM (TASM)*** is significantly faster and easier to use than the IBM Macro-Assembler (MASM). Whether you are new to assembly language and want to quickly write a small assembly language routine, or are an experienced MASM user tired of waiting months to assemble large files, ***TURBO EDITASM*** will bring the excitement back to assembly language.

***TURBO EDITASM* IS MUCH FASTER:**

- How fast is ***TASM?*** The graph below shows relative assembly times for a 48K source file. For large files like this we blow MASM's doors off at 3 times their speed. For smaller 8K files we positively vaporize them at 6 times their speed.

TASM	(110 sec.)
MASM	(340 sec.)

- ***TURBO EDITASM*** is faster for the following reasons: **(1)** Written entirely in assembly language (unlike MASM). **(2)** Editor, assembler and source file always in memory so you can go instantly from editing to assembling and back. **(3)** Eliminates the time needed to LINK programs. Executable COM files can be created directly. (Also creates OBJ files compatible with the IBM linker).

***TURBO EDITASM* IS EASIER TO USE:**

TASM includes many other features to make your programming simpler.

- Listings are sent directly to screen or printer. Assemblies can be single stepped and examined without having to leave the editor.
- Access the built-in cross reference utility from the editor.
- Full support of 186 and 286 (real mode) instructions.
- Both Microsoft and 8087 floating point formats are supported. 8087 and 287 instructions supported directly without macros for faster assembly.
- Calculator mode: Do math in any radix even using symbols from the symbol table.
- Direct to memory assembly feature lets you test execute your code from editor.
- Coming soon: A coordinated symbolic debugger.

COMPATIBILITY: ***TASM*** is source code compatible with MASM and supports macros, records and structures.

Introductory Price $49
With .OBJ Capability $99

Include $5.00 shipping and handling. California residents add 6% Sales Tax.

Dealer inquires welcome

Speedware™

916-988-7426
118 Buck Circle, Box D
Folsom, CA 95630

IBM, Microsoft trademarks of IBM Corp., Microsoft Corp.

Circle **no. 59** on reader service card.

Introducing SMK™, the

SEIDL MAKE UTILITY

SMK is an automatic software product generation utility for MS™-DOS. With SMK, you can update and track every module in your program. When changes are made to any module, SMK will recompile or reassemble changed modules and modules effected by the changes, and relink the modules in the proper order. SMK will work with any compiler, assembler and linker.

Advanced SMK Features Include:

- Proprietary dependency algorithm analyzes *all* dependencies before rebuilding *any* files.
- SMK understands complicated dependencies involving nested include files, source, and object code libraries.
- High-level dependency definition language makes setting up dependencies easy. Supports parameterized macros, local variables, constants, include files, command line parameters, line and block comments.
- Batch source code editor that allows automatic *source* file updating is included.
- Full pathname and directory support.
- Typeset user's manual and excellent error diagnostics make SMK easy to learn and easy to use.

SMK Introductory Price —
Save 40% — $84.00

SMK List Price (effective Jan. 86): $140.00
(include $3.50 postage & handling)

SEIDL COMPUTER ENGINEERING
1163 E. Ogden Ave., Suite 705-171
Naperville, IL 60540
(312) 983-5477

Circle **no. 114** on reader service card.

```
/*
 * ScsiRdWr -- do a driver read/write function.
 */
ScsiRdWr() {
        register struct ioParam *ip;
        long    len,part;
        short   blkno;
        char    *addr;

        save();

        ip = & Pbp->u.iop;
        part = Dp->dCtlPosition & ØxFFFFFEØØ;
        len = (ip->ioReqCount + 511) & ØxFFFFFEØØ;
        addr = ip->ioBuffer;
        while(len >= 512) {
                blkno = part>>9;
                if((Pbp->ioTrap & Øxff) == 2) {
                        ScsiRead(blkno, addr);
                 } else {
                        ScsiWrite(blkno, addr);
                }
                len -= 512;
                addr += 512;
                part += 512;
        }
        ip->ioActCount = ip->ioReqCount;

        restore();                      /* exit to I/O Done */
}
```

End Listings

FORTH

and Assembler for

65802/816

16 bit Processing for Upgraded 6502 Systems

Forth, Assembler, and Debugger for

65 Series Processors

On a VAX or PC*

Starlight Forth Systems

15247 N. 35th St
Phoenix, AZ 85032
(602) 992–5181

*65802 on Standard 6502 systems

OR

*65816 Access to 640K+ with Apple and

Checkmate Technology's New

MultiRam][e and MultiRam C Boards

Circle no. **142** on reader service card.

ST-FORTH $49

For your IBM PC/XT/AT/PCjr

A complete FORTH development system for beginners or experienced users

- 100% FORTH-83 Standard
- ALL source code provided
  Written entirely in FORTH & FORTH assembler
- Powerful, fast, easy-to-use editor
  Search, locate source, edit errors
- Extensive MS-DOS/PC-DOS interface
  Sequential & random files
- Many more useful utilities
- Supplied as MS-DOS .COM file
  Compatible with all versions of DOS
- 148-page User's Manual
- Unconditional money-back guarantee

Beginner's pkg based on Leo Brodie's *Starting FORTH* available

Call or write for brochure and order form.

Sunset Technology

1954 Menalto Ave.
Menlo Park, CA 94025
(415) 325-3680

Circle no. **144** on reader service card.

C64-FORTH/79
for the
Commodore 64/128
$69.95

- **C64-FORTH/79**™ integrated professional development environment.
- See our reviews in INFO 64, MIDNIGHT, and NY CBMUG. **C64-FORTH/79** is Commodore Approved.
- High performance 2D graphics extension including HRES multicolor line, circles, scaling, windowing, HRES character graphics, sprites, ram characters, 10 demo screens and more.
- Complete CBM compatible floating point package includes arithmetic, relational, SIN/COS, SQR, and more.
- Professional, 21 command, cursor screen editor with virtual memory, conditional macro assembler, and debug-decompiler facility.
- String extension for easy string processing.
- Compatible with CBM peripherals, popular third party peripherals and C64 operating setup/memory configurations.
- Easy to use 167 page manual written for the serious forth programmer with many examples, application screens, detailed command glossaries and compatible with "Starting Forth", or "Discover Forth."
- "SAVE TURNKEY" automatically compiles bootable turnkey application programs for royality free distribution.

(Commodore 64 and CBM are trademarks of Commodore)

TO ORDER
- Check, money order, bank card. COD's add $2.00.
- Add $4.00 postage and handling in USA and Canada.
- Mass. orders add 5% sales tax.
- Foreign orders add 20% shipping and handling.

PERFORMANCE MICRO PRODUCTS
P.O. Box 370
Canton, MA 02021
(617) 828-1209

Circle **no. 110** on reader service card.

WORD TECH SYSTEMS **dBIII** COMPILER

A Compiler For dBASE III
$600
Write-Hand Man
(Sidekick for CP/M)
8" Format $49.95 5¼" Formats $54.95

CP/M LIVES!
A die hard CP/M house actually exists.
We confess, we still sell, support, and use CP/M systems and software. Custom multiuser hardware and dBASE II applications are available.

d/MULTI
Multiuser dBASE II for TurboDOS
At last! File & record locking as easy as d-BASE-II. Now unlimited users can perform the magic of dBASE in the program or interactive mode. Don't look for it anywhere else — A MARTIAN EXCLUSIVE!
d/MULTI . . . $695
Dealer inquiries encouraged

Martian Technologies
8348 Center Dr., Ste. F. La Mesa, CA 92041
(619) 464-2924

Circle **no. 74** on reader service card.

The Magic Micro-Mainframe Data Editor

PIK'r selectively reformats mainframe data without re-keying.

PIK'r provides a full screen cut & paste environment, including mouse support.

You can pick the data you want from your reports, in the format you're familiar with, and put it all together for Lotus templates, dBASE, Multiplan, DIF, or word processors.

Takes data from any ASCII text file or report—from mainframe, mini, or micro. Works on any IBM-PC or compatible.

Review and combine several reports in one session, or produce several output files from one report.

Edit. Merge and divide columns. Automatically clean up numbers with embedded commas, leading dollars, trailing CR and parens—also supports European and metric formats. Transpose rows into columns, columns into rows.

$95 includes User Manual, Quick Start Card, full support services. Quantity and site discounts available.

SAMKHYA*/CORPORATION*
47 Sixth St, Suite 3000
PO Box 142
Petaluma, CA 94953
707-763-2800

Sales & Info: 800-442-0012 USA 800-442-5544 California

Circle **no. 20** on reader service card.

BRIEF™

"BRIEF, The Programmer's Editor, is simply the best text editor you can buy."
John Dvorak, InfoWorld 7/8/85

Choosing the right program editor is an extremely important decision. After all, more time is spent with your program editor than with any other category of software. A good program editor helps to stimulate creativity and produce programs of superior design. A poor program editor, on the other hand, can be worse than none at all. It can slow you down, get in your way, and make even the simplest tasks difficult.

BRIEF - PROGRAM EDITING *YOUR WAY*

Every programmer has an individual style that makes their work unique. Most program editors compromise that style by forcing the programmer to conform to their methods. Not so with BRIEF.

BRIEF's most powerful feature is its ability to conform to your way of programming. BRIEF can be easily tailored/customized to your individual preferences. For example, keyboard reassignment allows the keyboard to be used in whatever way you prefer. Keystroke macros allow long sequences of editor directives to be repeated by pressing a single key. Plus you can simultaneously work with numerous program and data files, and configure many windows to control BRIEF's visual presentation.

75% OF THE EXPERTS WHO HAVE TRIED BRIEF SWITCHED!

(Call For Details)

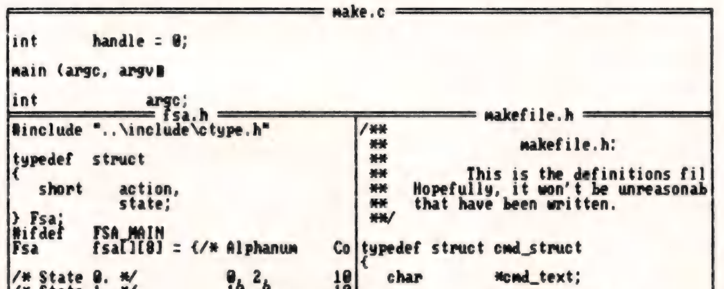
"A BONA FIDE UNDO" Steve McMahon – BYTE REVIEW "BUILD YOUR DREAM EDITOR" MARCH 1985

Here are Steve McMahon's exact words: "...BRIEF implements a true undo facility, by default allowing command-by-command recovery from the last 30 commands...The number of commands you want to be able to undo can be changed." (up to 300) "Only with BRIEF, though, was it possible to undo a macro that produced 4000 words of text with a single keystroke."

EVERY FEATURE YOU WANT

BRIEF supports practically every feature you've ever seen, and some you probably haven't thought of yet.

- Edit Multiple Large Files
- Windows (Tiled & Pop-up)
- Unlimited File Size
- Full UNDO (N Times)
- True Automatic Indent for C
- Exit to DOS Inside BRIEF
- Compile programs inside BRIEF
- Uses All Available Memory
- Intuitive Commands
- Tutorial
- Repeat Keystroke Sequences
- 15 Minute Learning
- Reconfigurable Keyboard
- Online Help
- Search for "Regular Expressions"
- Mnemonic Key Assignments
- Horizontal Scrolling
- Comprehensive Error Recovery
- And ... a Complete, Compiled, Programmable and Readable Macro Language.



```
══════════════════════════════ make.c ══════════════════════════════
int       handle = 0;

main (argc, argv

int                argc;
═══════════════ fsa.h ═══════════════════════ makefile.h ══════════
#include "..\include\ctype.h"        /**
                                      **            makefile.h:
typedef  struct                       **
{                                     **        This is the definitions fil
   short    action,                   **   Hopefully, it won't be unreasonab
            state;                    **   that have been written.
} Fsa;                                **/
#ifdef   FSA_MAIN
Fsa      fsa[][8] = {/* Alphanum    Co typedef struct cmd_struct
                                       {
/* State 0. */          0, 2,       10    char        *cmd_text;
/* State 1. */          10, 0,      10
/* State 2. */          0, 2,       1,    struct cmd_struct *next_cmd;
/* State 3. */          0, 5,       11 } *Cmd_Ptr, Cmd;
/* State 4. */          0, 4,       0,
Mismatched open parenthesis.                 Line: 11   Col: 17      2:17 pm
```

A TYPICAL BRIEF SCREEN

Notice there are three windows on the screen simultaneously and each one is viewing a different file. The mainline of a C program is visible in the uppermost window; the programmer has run a syntax checking macro which found a mismatched open parenthesis in the arguments to the mainline. The other two windows show header files containing information crucial to the design of the program. BRIEF can have an unlimited number of windows and files accessed simultaneously.

BRIEF'S ONLY LIMITATIONS ARE THE ONES *YOU* SET
Full refund if not satisfied in 30 days. ONLY $195

AVAILABILITY: IBM-PC and Compatibles, AT, & TANDY 2000

→ **FREE** with order: "Best of BRIEF Macros" — Contest entries include macros for Fortran, C, Calculator, Base Converter, etc. Call before November 30. **800-821-2492**

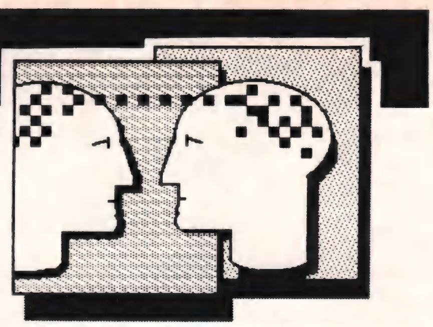
Solution Systems™

335-D Washington St., Norwell, MA 02061 (617) 659-1571 **800-821-2492.**

Steve McMahon's quote courtesy Suntype Publishing Systems. BYTE review by Mr. McMahon may be found in BYTE Magazine March 1985.

Circle no. 102 on reader service card.

CP/M EXCHANGE



by Bob Blum

The CP/M Exchange RCP/M system is available for your use 24 hours a day, 7 days a week. Reach it by dialing (404) 449-6588.

My queue runneth over. It seems that I always have far more projects underway than I will ever have time to complete. My workmates characterize my problem as simply not being able to manage my time efficiently. I know better, I know the real reason that this phenomenon constantly haunts me.

My time is being slowly pilfered away by the software developers who continue to write programs for CP/M. It's an insidious plot. What reasonable person would continue to support an outdated operating system such as CP/M and the archaic 8-bit microprocessors it runs on? The schemers know of my weakness for trying every new program that's advertised, and they use it to their best advantage. Some are even so bold as to send me their addicting wares of slavery without even letting me call to volunteer myself for human sacrifice.

So, as you can see, I'm not at fault for falling victim to my weaknesses.

All fun aside, more and more new 8-bit CP/M software is being introduced each day. I applaud all of those who have the foresight not to abandon a strong 8-bit market. By some standards I may be misguided, but I rest assured that I am in the best of company.

In the Queue

I'm anxiously waiting for my review copy of the latest single board computer from Micro Mint. This little barn burner, the size of a mini-floppy drive, sports the HR64180, the newest 8-bit chip from Hitachi. I'm told that this device is at least as powerful as an 8-Mhz Z80 and offers a much higher level of integration, including I/O ports and other features that normally require external devices. The most exciting part is its capability to address directly 512K of memory. My plans call for a review and then for porting CP/M Plus over to it, taking full advantage of all of its new advanced features. I think in a short time that a new level of performance will be established.

Even though DRI has decided to put CP/M Plus on the mature list, and will no longer actively support it, I am continuing to port it to different computers. My most recent adventure is with the Intercontinental 8-bit S-100 master board; on-board memory management and many other performance features make the ICM board a natural for CP/M Plus.

Among the top ranking features of CP/M Plus is its capability to time and date stamp files automatically at each access. This feature is especially important to me because I have the untidy habit of not religiously tracking my disk files during a heated development session. I wait for the project to conclude before trying to make good sense out of my files and then sometimes have trouble recalling the order in which I created them, and, most important, which is the most current. Until Plu*Perfect Systems began to sell DateStamper there was no software available to add this desirable feature to CP/M 2.2. I've been using DateStamper for several weeks and have found it a most useful tool and safeguard. I will be reviewing it in a future issue of this column.

Finally, my stack of things to do wouldn't be complete without a new programming tool. I have been using DSD80, a DDT-compatible debugger with many new and useful added functions. It's a very solid package that turns the drudgery of debugging into an art form. I will be reviewing it in the future.

Encore, Encore

After producing the fastest full featured Z80 assembler, SLR Systems could do nothing for an encore but bring out the fastest fully featured 8080 assembler.

SLRMAC shares all the operational features and configuration options of its family member Z80ASM. A total of 32 permanently configurable options combined with 20 run-time command line parameters make for the ultimate in flexibility and ease of use. I found several of the configuration options to be most interesting.

A maximum error threshold count can be set that automatically aborts the assembly when reached. This option precludes resetting your computer or in some other way manually aborting an assembly if excessive errors are present.

Another option dealing with errors sets a limit on the number of errors that can be displayed on the CRT before the display is frozen. It's no longer necessary to sit poised to strike the CTL-S key combination to stop the CRT display in an attempt to catch a glimpse of an error flying by at lightning speed. Instead, the display automatically stops upon reaching the set error count and waits for the operator's signal to continue.

One other option that I particularly like is that addresses on the listing can be generated in either logical order, high byte first, or the normal low byte-high byte fashion. The mental step of swapping the address bytes slows me down as I move around in a program listing.

Command line arguments allow almost all of the permanent configuration options to be overridden and the specification of many more options that are significant to an individual assembly.

One particularly important option deals with the type of binary file that is output from the assembly. The H option directs the output file to be in the most common .HEX format. To execute a program written in this format, of course, requires that it first be converted to a .COM file by processing it through the LOAD utility program. If the program consists of a single phase (all routines are self-contained) then another option can be used that will dramatically speed the process of converting the source program to an executable .COM file. By including the A option on the command line, SLRMAC will assemble the program in a single pass through the source file and create a .COM output file ready for execution. This option will also work for programs that contain forward references. It seems that SLR went the extra mile and included forward reference resolution logic in the assembler so that a single pass assembly is now a workable and timesaving reality.

If you're a modular programmer, as I am, then the three options dealing with relocatable output files will be of keen interest to you. Two are variations of the pseudo-standard Microsoft naming convention, while the other permits the design maximum of seven characters to be utilized. To provide advanced features not otherwise available in other binary formats, an SLR format can also be selected. This format is not only more compact than the others but it also provides for a full 16-character external name. This increased name size has proven to be very valuable to me in extremely complex systems containing many hundreds of individual modules, not to mention how much more quickly the link editing operation runs.

The restrictions pertaining to statement labels found in most assemblers have been removed from SLRMAC in most cases. It's not at all uncommon in some assemblers to be forced to follow all labels with a colon. In others, the archaic practice of requiring that only certain types of statements have labels followed by a colon is followed. SLRMAC doesn't require that any label be followed by a colon except in the case where a duplication exists between a reserved word or a macro name. In all other instances the use of the colon is entirely optional. One final word about labels in SLRMAC: a label of any length not exceeding the maximum line length of 128 characters is permitted and the beginning 16 characters are significant.

Many problems can occur when passing relocatable expressions from program to program. Most assemblers only allow the most simple uses of this type of expression. For instance, when loading the external value into a register, any type of math involving this type of expression is prohibited, which many times requires that a great deal of extra code be written. SLRMAC allows full expression usage via a special relative data type. In this way, arithmetic operations involving multiple relocatable expressions can be resolved properly at link editing time. But a word of caution is in order; the relative data type is only accepted by SLR's own link editing program.

DEBUG FAST!

As a professional, your time is valuable. Advanced Trace86™ is designed as a debugging environment to increase productivity. In many situations, Advanced Trace 86™ can pay for itself in one or two debugging sessions.

Advanced Trace86™
FEATURES

- Trace Screen with user-configurable windows for disassembled code, registers, flags, the stack, the 8087 registers, and memory.
- Single-step tracing with virtually unlimited **back tracing.**
- In-line assembly capabilities, edit .EXE and .COM files in memory (including non-destructive line insertion for .COM files).
- Symbolic debugging capabilities - read or create line labels and comments, read variables.

Debugger Comparison Chart - Selected Features

Features	AT86	Code.	Per.
1. Load Program as memory resident (like Sidekick)	X		X
2. IBM Professional DEBUG & Periscope boards supported	X		X
3. Keyboard break-out support - Ctrl-Enter key combo	X	X	
4. Conditional breakpoints	X		X
5. Hex/Decimal calculator & converter	X		X
6. Convert numbers to binary	X		
7. ASCII chart (pop up)	X		
8. In-line Assembler with code insertion capability	X	X	
9. 80286/80287 support in Assembler/Disassember	X		
10. Protect Virtual Mode Support for 80286	X		
11. 8087/80287 window - registers in both decimal & hex	X		
12. Display memory in ASCII, byte, word, & double word	X		X
13. Define "memory structures" for display/editing	X		
14. Try out DOS interrupts command	X		
15. Pop-up Help windows	X		
16. Command macros defined & saved to disk	X		
17. DOS TYPE, DIR, & ERASE commands	X		
18. Search memory for Assembler code - multiple instructions	X		
19. Execute/trace program in reverse & restore machine state	X		

AT86 - Advanced Trace86 **Code.** - Codesmith **Per.** - Periscope

Requires: IBM PC or compatible, minimum 128K RAM. Retail Price - $175.00

Morgan Computing Co., Inc.
P.O. Box 112730 Carrollton, Texas 75011 (214) 245-4763

Circle no. **128** on reader service card.

The standard program counter maintenance pseudo-ops are available in SLRMAC. These include ORG, CESG, DSEG, ASEG, and COMMON, to name a few. A modified form of COMMON is available to assign up to 5 additional address spaces that can be assigned beginning addresses at link editing time. This can be very important if the program being written is targeted for a machine control environment in which dedicated portions of the address space must be defined. Again, this is an extension to the standard binary output files and as such is only available in the SLR format relocatable file.

When defining data storage areas it is not at all uncommon to need a string delimiter placed at the end of, for example, a console message. The DEFZ data definition statement will automatically generate a byte containing binary zeros at the end of the quoted string. And the DC statement will automatically turn bit 7 on in the last byte of the defined string.

A full complement of conditional assembly pseudo-ops and a full macro capability are provided. Two pseudo-ops that I have found to be especially useful are IFIDN and IFDIF. IFIDN tests for equality between two strings and sets the true condition according to the result. IFDIF performs exactly the opposite function of testing for inequality between the two strings and sets the true condition according to the result.

The method used to process the MACLIB pseudo-op is unique and worthy of comment. Intel mnemonic codes were designed to be used with the 8080, and there is no provision made for the additional instructions available on the Z80. When the program is being written for the Z80 and full use of its instruction set is desired, the extended instruction mnemonics are usually added to the assembly through a macro library. This solves the problem, but at the expense of slower run time and the necessity to maintain another macro library. SLRMAC, on the other hand, directly recognizes the extended set of Z80 mnemonics. Upon encountering the MACLIB Z80 statement the extend-

Thinking of the C Language?

THINK COMPUTER INNOVATIONS

NEW!! **C86 VERSION 2.3 with Source Level Debugging Support**

The C language has rapidly become the development language of choice for applications ranging from Operating Systems to Accounting Packages. WHY? Its structured approach and extreme portability make it perfectly suited to today's fast-paced environment.

Of all of the C Compilers available for PC/MSDOS, more programmers choose COMPUTER INNOVATIONS' C86. WHY? Because it's part of a COMPREHENSIVE family of C products with an unparalleled reputation for performance, reliability, and stability.

C86 2.3 C COMPILER

C for PC/MSDOS began with C86 and today it remains perhaps the most solid, stable C Compiler available. Even competitor's ads show C86 as a consistent top level performer in benchmark testing.

Version 2.3 offers a host of new features including source level debugging support and a 40% boost in compilation speed. Call for complete specifications.

COST: $395 UPDATE TO 2.3: $35 w/old diskettes NOT COPY PROTECTED
CALL ABOUT VOLUME DISCOUNTS

LEARN C INTERACTIVELY WITH INTRODUCING C

Intimidated by rumors about the difficulty of learning C? Need to train your staff quickly? INTRODUCING C can help. INTRODUCING C combines a thorough, self-paced manual with a unique C interpreter to provide a fast, efficient method of learning C. Designed for both professional and casual programmers, it provides a comprehensive understanding of important C concepts such as standard K&R syntax and operators, full structures and unions, arrays, pointers, and data types. Requires IBM PC, XT, or AT with one disk drive and 192K bytes of memory.

COST: $125 - NOT COPY PROTECTED

CI PROBE SOURCE DEBUGGER

Take advantage of C86 2.3 source level debugging support with CI PROBE. Cut down program development time and save money! CI PROBE is highly economical yet has the features of debuggers costing far more.

COST: $225 - NOT COPY PROTECTED

C-TERP C86 COMPATIBLE INTERPRETER

The C-TERP INTERPRETER is a full K&R implementation that allows you to write code and execute it immediately without the compile and link steps. Once you have your program running with C-TERP you can compile the code (without alterations) with C86 for fast, efficient executable files. C-TERP requires 256K, 512K is recommended.

COST: C86 version - List Price: $300, Special Computer Innovations Price $250.
Combined C86 & Lattice version - List Price: $400, Special Computer Innovations Price $350.

Start With Us, Stay With Us

Computer Innovations offers a complete range of products that let you enter the C environment and create applications with the most advanced set of development tools available. Unparalleled tech support assures that you're always at the height of productivity.

To order call: 800-921-0169

COMPUTER INNOVATIONS, INC.

980 Shrewsbury Ave., Tinton Falls, NJ 07724 • (201) 542-5920

C-TERP is a trademark of Gimple Software. Prices and specifications subject to change without notice.

Circle **no. 96** on reader service card.

ed set of mnemonics are simply turned on, resulting in no speed degradation due to the loading of a macro library and macro generation.

SLRMAC is another notable product from SLR Systems. It not only does the job that it's advertised to do but it is also free of the many petty annoyances and inaccuracies that plague so many similar products. Its many advanced features, ease of use, and blinding speed make it a must for anybody in need of an Intel mnemonic assembler.

Your Attention Please

Before progressing further with the discussion of the Ampro Little Board computer begun last month, a few words on Z80 interrupts are in order. The Z80 and its family of related peripheral support chips have an excellent built-in interrupt system. Each support chip that has more than one active element has its own internal interrupt priority scheduler and the capability to be daisy-chained into a system. This structure does not require assistance from interrupt priority schedulers in all but the biggest systems.

The main controlling element in the system, the Z80 CPU, can handle an interrupt in one of three ways depending on the setting of the interrupt mode. This fact permits the software to determine, to a large degree, how complex the interrupt system will be.

The most basic interrupt mode, mode 0, operates identically to that of the 8080 interrupt response mode and was included to provide full compatibility with that earlier and less powerful CPU. This mode is set by executing the IM 0 instruction. With this mode, the interrupting device can place any instruction on the data buss and the CPU will execute it. Thus, the interrupting device, not the memory, supplies the next instruction to be executed.

Mode 1, when enabled by the IM 1 instruction, causes the CPU to execute a restart to memory location 38h in response to an interrupt. This action is identical to that of the nonmaskable interrupt except the call location in memory is 38h instead of 66h.

Finally, mode 2, the most powerful interrupt mode, combines three system elements to resolve the effective memory address loaded into the program counter. With a single 8-bit value supplied by the interrupting peripheral device an indirect call can be made to any memory location.

With this mode the CPU forms a 16-bit memory address. The upper eight bits are supplied from the Z80 interrupt register, and the lower eight bits are supplied by the highest priority interrupting peripheral device. This 16-bit address is used as an indirect memory address pointing to a stored table of pointer words that point to the processing routine responsible for servicing the interrupt. Actually, only seven bits are required from the interrupting device because bit 0 is always zero. This structure might seem overly complex until you consider that through software the service routine tables can be changed at will, providing the ultimate in flexibility.

Interrupts and the Little Board

The Ampro Little Board computer satisfied all the requirements I set for my RBBS system except the need for a real-time clock. My first impulse was to attach one through a serial port, but I soon remembered that both ports were busy; one with the CRT and the other with the modem. The lack of any unused I/O ports stymied any chance I had of using an external clock. The only option open to me was to look to the Little Board for a solution.

Digging into the schematics of the Little Board revealed a fairly straightforward solution to my problem. The engineers responsible for designing the Little Board used the Z80 and its related support chips exclusively to minimize interface logic requirements and arranged them to allow full use of their daisy-chained interrupt capabilities. So it seemed that all I needed was to find an unused timer, or some other source of regular interrupts, and a little software to implement a real-time clock.

Fortunately, my search was a short one. Channel 2 of the Clock Timer Circuit (CTC), used for generating the baud rates for the serial ports, was left totally unused. All that was needed to start it working as a real-time clock was the few lines of code of Listing One (page 120).

My first task was to define a table of pointer values (section 2 in the listing) addressing the interrupt processing routines. The address of the first entry of this table would later be loaded into the interrupt vector register. Following this table are the processing routines. I chose to place the clock values in low memory locations: 08h hours, 09h minutes, 0Ah seconds, and 0Bh tenths of seconds.

Next, I had to insure that the constant flow of interrupts from the CTC would not adversely affect any other time-sensitive system component such as the floppy disk controller. The only positive way to do this was to disable the interrupts (sections 3 through 6 in the listing) prior to entering the data transfer routines. In practice, this proved to be absolutely necessary because of excessive disk errors when running with interrupts enabled.

The side effect of this, however, is to destroy the clock's accuracy. Since it's unknown exactly how long the clock will be stopped during a disk data transfer, it's impractical to add in a fudge factor upon exiting the disk routines. By comparing the computer clock against a stopwatch I found that during a highly disk intensive period the computer clock would run approximately 50% slow. Under more normal circumstances the clock wasn't accurate, but was close enough for my purpose of timing how long a user was on the RBBS.

To achieve my goal of at least 95% accuracy I would have to run the floppy disk controller with interrupts enabled, but, as I already pointed out, that produces an unbearable error rate. My only choice was to modify the hardware slightly to allow both the CTC and the floppy disk controller to run with interrupts enabled. That is a project I am currently working on. Once complete, I will publish the software and other modifications in this column.

Getting back to the real-time clock

Finally. BSW-Make.

The Boston Software Works now brings a complete implementation of the Unix "make" facility to MS-DOS. No more recompiling every file in sight after a small edit; no more wondering if you've really rebuilt every module affected by an edit. Just type "make" and BSW-Make automatically builds your product quickly, efficiently and correctly.

BSW-Make supports:

- most compilers and assemblers
- MS-DOS or PC-DOS v2.00 or later
- macros for parameterized builds
- default rules
- MS-DOS pathnames
- any MS-DOS machine (192K minimum)

Only $69.95 postpaid (Mass. residents add 5% sales tax)

The Boston Software Works
120 Fulton Street, Boston, MA 02109
(617) 367-6846

Circle **no. 126** on reader service card.

u4th

THE UNIX/XENIX-compatible Forth:

- C primitives
- dynamic memory management
- direct-threaded
- UNIX interfaces
- object-oriented Forth-source included!
- 400-page manual and glossary
- no royalties for developers

New Low Prices!
Plexus, Sun, Intel 286: $495.00
PC XT, AT, Tandy: $195.00
Now! VAX, AT&T 3B

UBIQUITOUS SYSTEMS
13333 BEL-RED ROAD NE
BELLEVUE, WA 98005
206-641-8030

UNIX(TM) AT&T XENIX(TM) MICROSOFT

Circle **no. 152** on reader service card.

AT LAST —
A handbook that contains the Programming Codes for 100's of popular printers.
Announcing:

PROGRAMMERS' HANDBOOK OF COMPUTER PRINTER COMMANDS

The handbook gives you:

- Codes for printers made by over 40 Printer Manufacturers.
- Easy to use spiral bound book of over 250 pages of Programming Codes written in table form.
- Codes arranged by Written Code, Hex and Decimal equivalent, and with a brief description of what each code does.
- Codes for either Daisy-Wheel or Dot Matrix Printers (models through 1984).

ONLY $37.95 + $2 shpg./hdlg. ppd. with a two week approval guarantee. IF NOT SATISFIED, return in original carton for refund of book price only.

FOR MORE INFORMATION OR TO ORDER CALL OR WRITE:

Cardinal Point
INCORPORATED
(812) 876-7811 (9-5 EST)
P.O. BOX 596, ELLETTSVILLE, IN 47429
We accept MC, VISA, MO—same day shpg. COD—$2 extra. CKs—Allow extra 14 days.

Circle **no. 11** on reader service card.

C Users' Group

Over 45 volumes of public domain software including:

- compilers
- editors
- text formatters
- communications packages
- many UNIX-like tools

Write or call for more details

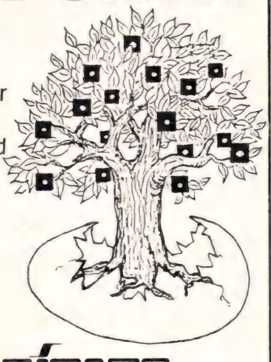
The C Users' Group
415 E. Euclid • Box 97A
McPherson, KS 67460
(316) 241-1065

Circle **no. 17** on reader service card.

Tree Shell

A Graphic Visual Shell for Unix/Xenix End-Users and Experts Alike!

Dealer inquiries welcomed.

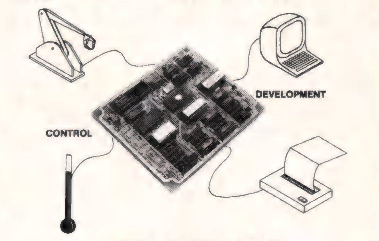


COGITATE

"A Higher Form of Software"
24000 Telegraph Road
Southfield, MI 48034
(313) 352-2345
TELEX: 386581 COGITATE USA

Circle **no. 24** on reader service card.

SBC88 DEVELOPEMENT AND CONTROL STYSTEM



WRITE IT — RUN IT — ROM IT

A single board computer development and control system that is so simple to use, you will be developing applications programs the first day!

- Choice of Basic or FORTH in ROM
- Onboard EPROM programmer for complete program development
- 8 channel, 8 bit analog to digital converter
- RS-232 terminal and parallel printer port for program entry
- Two 8 bit input ports Two 8 bit output ports
- 7 current sinking outputs rated at 500 mA, 50 VDC
- Time of day
- Up to 32 K of user memory
- 8088 16 bit uP
- **Floating point UNIFORTH available**

Units start at $279 ($99 1000)
Compatible power supplies start at $49

Vesta Technology, Inc. 7100 W. 44th Ave. Suite 101
Wheat Ridge, CO 80033 (303) 422-8088

Circle **no. 156** on reader service card.

TURBO PROFESSIONAL™

Turbo Pascal Tools

- SERVICE INTERRUPTS IN PASCAL
- EASY SIDEKICK™-LIKE PROCEDURES
- VARIABLE SIZE WINDOWS
- KEYBOARD MACROS
- DOS PROGRAM EXECUTION & RETURN
- 11 INCLUDE FILES, 109 ROUTINES
- SOURCE CODE INCLUDED

SUPER MACS™ example program features keyboard macros, save macro files, load macro files, movable window, write screen to ASCII file, print concurrently in DOS 3.0 or greater. Works within other programs. Available separately as .COM file.

SUPER MACS ONLY (.COM)	-$10.00
WITH TURBO PROFESSIONAL & SUPER MACS SOURCE CODE	-$49.95
S & H	-$ 5.00
MC & VISA orders:	(206) 367-0650

Or send check or money order to:
Sunny Hill Software
13732 Midvale North, Suite 206
Seattle, WA 98133

Requires IBM PC, XT, AT, or 100% compatible; DOS 2.0 - 3.1; Turbo Pascal 2.0 - 3.01B for compatibles. Sidekick trademark Borland Intl.

Circle **no. 10** on reader service card.

No source code for your REL files?

REL/MAC

converts a REL file in the Microsoft™ M80 format to an 8080 or ZILOG™ Z80 source code MAC file with insertion of all public and external symbols.

- **REL/MAC** makes MAC source files
- **REL/MOD** lists library modules
- **REL/VUE** displays the bit stream
- **REL/PAK** includes all of the above
- 8080 REL/MAC **demo disk** $10.00

REL/PAK for 8080 only	$99.95
REL/PAK for Z80 & 8080	$134.95

on 8" SSSD disk for CP/M™ 2.2

Send check, VISA, MC or C.O.D. to

MICROSMITH COMPUTER TECHNOLOGY
P.O. BOX 1473, ELKHART, IN 46515
1-800-622-4070
(Illinois only 1-800-942-7317)

Circle **no. 41** on reader service card.

ICs PROMPT DELIVERY!!!

SAME DAY SHIPPING (USUALLY)

OUTSIDE OKLAHOMA: NO SALES TAX

640 Kbyte MOTHERBOARD KITS: IBM PC/XT, Zenith 150 & 100, Compaq Portable & Plus: $69.50

QUANTITY ONE PRICES SHOWN

8087-2 COPROCESSORS $125.00

DYNAMIC RAM			
256K	256Kx1	120 ns	$ 3.75
256K	256Kx1	150 ns	2.75
128K	128Kx1	120 ns	6.75
64K	16Kx4	150 ns	2.75
64K	64Kx1	150 ns	1.10
EPROM			
27C256	32Kx8	250 ns	$12.75
27256	32Kx8	250 ns	6.00
27128	16Kx8	250 ns	3.30
27C64	8Kx8	200 ns	5.25
2764	8Kx8	250 ns	2.75
2732A	4Kx8	250 ns	2.75
STATIC RAM			
6264LP-15	8Kx8	150 ns	$4.75

OPEN 6½ DAYS: WE CAN SHIP VIA FED-EX ON SAT

ORDERS BY Th: Std Air Fr: P-One NO EXTRA COST ON F-EX SAT DELIVERY

MasterCard/VISA or UPS CASH COD
Factory New, Prime Parts µP∞
MICROPROCESSORS UNLIMITED
24,000 S. Peoria Ave., BEGGS, OK. 74421 **(918) 267-4961**

Prices shown above are for September 2, 1985

Please call for current prices. Prices subject to change. Please expect higher or lower prices on some parts due to supply & demand and our changing costs. Shipping & insurance extra. Cash discount prices shown. Orders received by 6 PM CST can usually be delivered to you **by the next morning, via Federal Express Standard Air @ $6.00, or Priority One @ $15.00!**

Circle **no. 64** on reader service card.

routines, the last addition I had to make to the BIOS (sections 7 and 8 in the listing) was to initialize the CTC and the interrupt register with the proper constants at cold boot time.

If you are inclined to make these changes to your machine, be sure that you are running version 2.3 of the BIOS, also known as the hard disk BIOS. Have a CTC reference manual handy to use when setting initial values.

One final comment: My system has a hard disk that ports into the Little Board through a SCSI peer buss adapter. I have found that this configuration runs completely error free because the hard disk controller is a buffered device able to transfer data while other tasks are active.

DDJ

Reader Ballot
Vote for your favorite feature/article.
Circle Reader Service **No. 195.**

CP/M Exchange Listing

(Text begins on page 114)

```
1) Add at line number 551 after statement FDALV   EQU   50

INTRPT  SET     YES                 ; SET INTERRUPTS TO NO AS DEFAULT
TIMER   EQU     YES                 ; USE CHANNEL 2 OF CTC AS A TIMER
TYPER   EQU     NO                  ; USE BUFFERED KEYBOARD

2) Add at line number 1028 after statement WTBLEN  EQU  $-WINCH3

        IF      TIMER

;============================================================
;
;       MODE 2 INTERRUPT VECTORS FOR THE CTC
;
;============================================================

        ORG     ($+17) AND 0FFF0H

TIMER$INT$TABLE:

        DW      BAUD$A
        DW      BAUD$B
        DW      TIMER$RTN
        DW      DSK$INT
        ENDIF

TIMER$RTN:

        PUSH    PSW                 ;SAVE THOSE REGISTERS THAT WE DESTROY
        PUSH    H                   ;*

        LXI     H,12                ;POINT AT HUNDREDS
        INR     M                   ;INCREMENT IT
        MOV     A,M                 ;GET HUNDREDS DIGIT
        SUI     10                  ;AT LIMIT
        JNZ     TIMER$EXIT          ;NO
        MOV     M,A                 ;ZERO HUNDREDS

        DCX     H                   ;POINT AT TENS
        INR     M                   ;INCREMENT IT
        MOV     A,M                 ;GET TENS DIGIT
        SUI     10                  ;AT LIMIT
        JNZ     TIMER$EXIT          ;NO
        MOV     M,A                 ;ZERO TENS

        DCX     H                   ;POINT AT SECONDS
        INR     M                   ;INCREMENT IT
        MOV     A,M                 ;GET SECOND DIGIT
        SUI     60                  ;AT LIMIT
        JNZ     TIMER$EXIT          ;NO
        MOV     M,A                 ;ZERO SECOND

        DCX     H                   ;POINT AT MINUTES
```

(Continued on page 122)

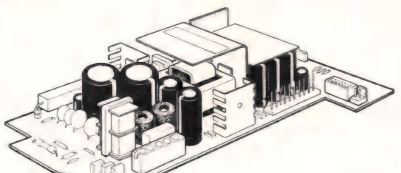
ALL SALES ARE MADE SUBJECT TO THE TERMS OF OUR 90 DAY LIMITED WARRANTY. A COPY OF THIS WARRANTY IS AVAILABLE FREE, ON REQUEST.

DIGITAL RESEARCH COMPUTERS

(214) 225-2309

Switching Power Supply!

+ 5VDC - 8 Amps
+12VDC - 5 Amps
-12VDC - 1 Amp
(81 WATTS MAX)



$29.95 ea.

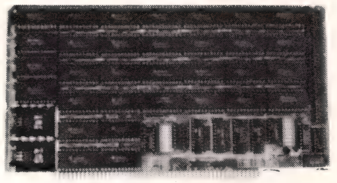
4 FOR $99

ADD $2 EA. UPS

BRAND NEW UNITS, MFG. BY BOSCHERT FOR HEWLETT PACKARD! PERFECT FOR SMALL COMPUTER AND DISK DRIVE APPLICATIONS #XL81-5630. INPUT 110V/220V, 50/60 HZ. NOMINAL OUTPUTS: +5VDC @ 8A, +12VDC @ 5A, -12VDC @ 1A. TOTAL MAX OUTPUT. MUST BE LIMITED TO 81 WATTS TOTAL! (WITH PIN OUT SHEET.) ORIGINAL FACTORY BOXED.

64K S100 STATIC RAM

$119.00 KIT



LOW POWER!
150 NS ADD $10

BLANK PC BOARD WITH DOCUMENTATION
$49.95

SUPPORT ICs + CAPS
$17.50

FULL SOCKET SET
$14.50

FULLY SUPPORTS THE NEW IEEE 696 S100 STANDARD (AS PROPOSED)

FOR 56K KIT $105

ASSEMBLED AND TESTED ADD $50

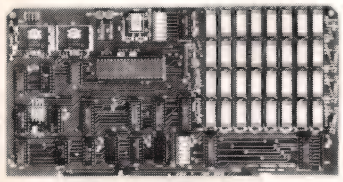
PRICE CUT!

FEATURES:
- Uses new 2K x 8 (TMM 2016 or HM 6116) RAMs.
- Fully supports IEEE 696 24 BIT Extended Addressing.
- 64K draws only approximately 500 MA.
- 200 NS RAMs are standard. (TOSHIBA makes TMM 2016s as fast as 100 NS. FOR YOUR HIGH SPEED APPLICATIONS.)
- SUPPORTS PHANTOM (BOTH LOWER 32K AND ENTIRE BOARD).
- 2716 EPROMs may be installed in any of top 48K.
- Any of the top 8K (E000 H AND ABOVE) may be disabled to provide windows to eliminate any possible conflicts with your system monitor, disk controller, etc.
- Perfect for small systems since BOTH RAM and EPROM may co-exist on the same board.
- BOARD may be partially populated as 56K.

256K S-100 SOLID STATE DISK SIMULATOR!

WE CALL THIS BOARD THE "LIGHT-SPEED-100" BECAUSE IT OFFERS AN ASTOUNDING INCREASE IN YOUR COMPUTER'S PERFORMANCE WHEN COMPARED TO A MECHANICAL FLOPPY DISK DRIVE.

PRICE CUT!



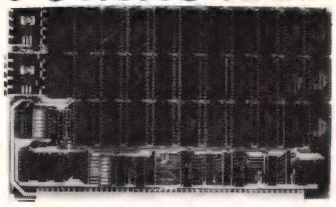
FEATURES:
- 256K on board, using + 5V 64K DRAMS.
- Uses new Intel 8203-1 LSI Memory Controller.
- Requires only 4 Dip Switch Selectable I/O Ports.
- Runs on 8080 or Z80 S100 machines.
- Up to 8 LS-100 boards can be run together for 2 Meg. of On Line Solid State Disk Storage.
- Provisions for Battery back-up.
- Software to mate the LS-100 to your CP/M* 2.2 DOS is supplied.
- The LS-100 provides an increase in speed of up to 7 to 10 times on Disk Intensive Software.
- Compare our price! You could pay up to 3 times as much for similar boards.

BLANK PCB (WITH CP/M* 2.2 PATCHES AND INSTALL PROGRAM ON DISKETTE)
$69.95
(8203-1 INTEL $29.95)

(ADD $50 FOR A&T) $149.00
#LS-100 (FULL 256K KIT)

64K SS-50 STATIC RAM

$99.95 (48K KIT)



LOW POWER!
RAM OR EPROM!

BLANK PC BOARD WITH DOCUMENTATION
$52

SUPPORT ICs + CAPS
$18.00

FULL SOCKET SET
$15.00

56K KIT $109
64K KIT $119

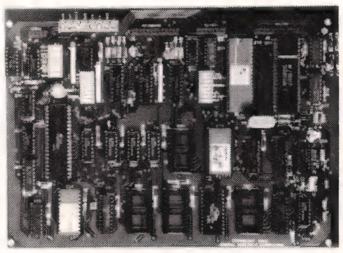
ASSEMBLED AND TESTED ADD $50

FEATURES:
- Uses new 2K x 8 (TMM 2016 or HM 6116) RAMs.
- Fully supports Extended Addressing.
- 64K draws only approximately 500 MA.
- 200 NS RAMs are standard. (TOSHIBA makes TMM 2016s as fast as 100 NS. FOR YOUR HIGH SPEED APPLICATIONS.)
- Board is configured as 3-16K blocks and 8-2K blocks (within any 64K block) for maximum flexibility.
- 2716 EPROMs may be installed anywhere on Board.
- Top 16K may be disabled in 2K blocks to avoid any I/O conflicts.
- One Board supports both RAM and EPROM.
- RAM supports 2MHZ operation at no extra charge!
- Board may be partially populated in 16K increments.

THE NEW ZRT-80 CRT TERMINAL BOARD!

A LOW COST Z-80 BASED SINGLE BOARD THAT ONLY NEEDS AN ASCII KEYBOARD, POWER SUPPLY, AND VIDEO MONITOR TO MAKE A COMPLETE CRT TERMINAL. USE AS A COMPUTER CONSOLE, OR WITH A MODEM FOR USE WITH ANY OF THE PHONE-LINE COMPUTER SERVICES.

FEATURES:
- Uses a Z80A and 6845 CRT Controller for powerful video capabilities.
- RS232 at 16 BAUD Rates from 75 to 19,200.
- 24 x 80 standard format (60 Hz).
- Optional formats from 24 x 80 (50 Hz) to 64 lines x 96 characters (60 Hz).
- Higher density formats require up to 3 additional 2K x 8 6116 RAMS.
- Uses N.S. INS 8250 BAUD Rate Gen. and USART combo IC.
- 3 Terminal Emulation Modes which are Dip Switch selectable. These include the LSI-ADM3A, the Heath H-19, and the Beehive.
- Composite or Split Video.
- Any polarity of video or sync.
- Inverse Video Capability.
- Small Size: 6.5 x 9 inches.
- Upper & lower case with descenders.
- 7 x 9 Character Matrix.
- Requires Par. ASCII keyboard.



$89.95 #ZRT-80
(COMPLETE KIT, 2K VIDEO RAM)

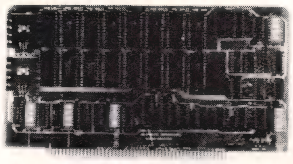
BLANK PCB WITH 2716 CHAR. ROM. 2732 MON. ROM
$49.95
SOURCE DISKETTE - ADD $10
SET OF 2 CRYSTALS - ADD $7.50

FOR 8 IN. SOURCE DISK (CP/M COMPATIBLE) ADD $10

32K S100 EPROM/STATIC RAM

FOUR FUNCTION BOARD!

NEW! NEW!



EPROM II
FULL EPROM KIT
$69.95

A&T EPROM ADD $35.00

BLANK PC BOARD WITH DATA
$39.95

SUPPORT IC'S PLUS CAPS
$16

FULL SOCKET SET
$15

We took our very popular 32K S100 EPROM Card and added additional logic to create a more versatile EPROM/RAM Board.

PRICES SLASHED!

FEATURES:
- This one board can be used in any one of four ways:
  A. As a 32K 2716 EPROM Board
  B. As a 32K 2732 EPROM Board (Using Every Other Socket)
  C. As a mixed 32K 2716 EPROM/2K x 8 RAM Board
  D. As a 32K Static RAM Board
- Uses New 2K x 8 (TMM2016 or HM6116) RAM's
- Fully Supports IEEE 696 Buss Standard (As Proposed)
- Supports 24 Bit Extended Adressing
- 200 NS (FAST!) RAM'S are standard on the RAM Kit
- Supports both Cromemco and North Star Bank Select
- Supports Phantom
- On Board wait State Generator
- Every 2K Block may be disabled
- Addressed as two separate 16K Blocks on any 64K Boundary
- Perfect for MP/M* Systems
- RAM Kit is very low power (300 MA typical)

32K STATIC RAM KIT — $99.95
For RAM Kit A&T — Add $40

Digital Research Computers

P.O. BOX 461565 • GARLAND, TEXAS 75046 • (214) 225-2309

TERMS: Add $3.00 postage. Orders under $15 add 75¢ handling. No C.O.D. We accept Visa and MasterCharge. Tex. Res. add 5-1/8% Tax. Foreign orders (except Canada) add 20% P & H. Orders over $50 add 85¢ for insurance.

*TM OF DIGITAL RESEARCH INC. (CALIF.)

WE ARE NOT ASSOCIATED WITH DIGITAL RESEARCH INC. (CALIF.) THE SUPPLIERS OF CPM SOFTWARE

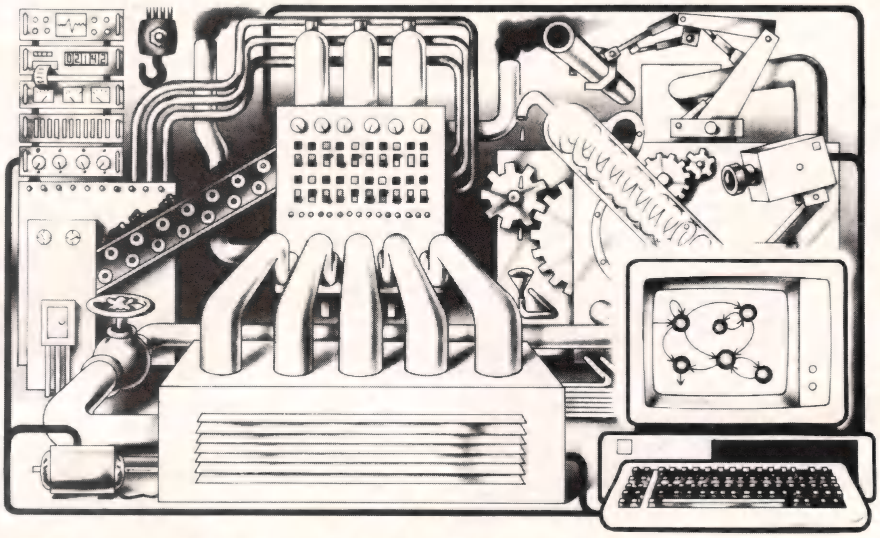
Circle **no. 33** on reader service card.

CP/M Exchange Listing

(Listing continued, text begins on page 114)

```
        INR     M                   ;INCREMENT IT
        MOV     A,M                 ;GET MINUTE DIGIT
        SUI     60                  ;AT LIMIT
        JNZ     TIMER$EXIT          ;NO
        MOV     M,A                 ;ZERO MINUTE

        DCX     H                   ;POINT AT HOURS
        INR     M                   ;INCREMENT IT
        MOV     A,M                 ;GET HOUR DIGIT
        SUI     24                  ;AT LIMIT
        JNZ     TIMER$EXIT          ;NO
        MOV     M,A                 ;ZERO HOUR

TIMER$EXIT:

        POP     H
        POP     PSW

BAUD$A:
BAUD$B:
DSK$INT:

        EI
        RETI

3) Add at line number 3144 after label WR:

        IF      INTRPT
        DI
        ENDIF
```

Csharp Realtime Toolkit



Realtime on MSDOS? Csharp can do it! Get the tools without operating system overhead. Cut development time with C source code for realtime data acquisition and control. **Csharp** includes: graphics, event handling, procedure scheduling, state system control, and interrupt handling. Processor, device, and operating system independent. **Csharp** runs standalone or with: MSDOS, PCDOS, or RT11. **Csharp** runs on: PDP-11 and IBM PC. **Csharp** includes drivers for Hercules and IBM graphics boards, Data Translation and Metrabyte iO boards, real time clock, and more. Inquire for Victor 9000, Unix, and other systems. Price: $600

SYSTEMS GUILD

Systems Guild, Inc., P.O. Box 1085, Cambridge, MA 02142
(617) 451-8479

Circle **no. 162** on reader service card.

4) Add at line number 3159 after label WREXIT:

```
        IF      INTRPT
        EI
        ENDIF
```

5) Add at line number 3199 after label RD:

```
        IF      INTRPT
        DI
        ENDIF
```

6) Add at line number 3214 after label RDEXIT:

```
        IF      INTRPT
        EI
        ENDIF
```

7) Add at line number 4431 before label LOGMSG: and before JMP GOCPM.

```
        IF      TIMER
        XRA     A                   ;GET A ZERO
        STA     8                   ;CLEAR TIMER SAVE AREAS
        STA     9                   ;*
        STA     10                  ;*
        STA     11                  ;*
        STA     12                  ;*
        MVI     A,(LOW TIMER$INT$TABLE) AND 0F0H
        OUT     CTCA0               ;SET INTERRUPT VECTOR LOW 8 BITS
        MVI     A,HIGH TIMER$INT$TABLE
        STAI                        ;SET INTERRUPT VECTOR HIGH 8 BITS
        IM2                         ;SET INTERRUPT MODE 2
        MVI     A,0A7H              ;TIMER MODE - 256 PRESCALER - CONSTANT
                                    ;FOLLOWS - RESET - CONTROL
        OUT     CTCA2
        MVI     A,156               ;TIME CONSTANT
        OUT     CTCA2
        ENDIF
```

8) Add at line number 4458 and after label LOGMD:

```
        IF      TIMER
        DB      CR,LF,'Real Time Clock Active'
        ENDIF
```

End Listing

"C/80 . . . the best software buy in America!"

—MICROSYSTEMS

Now available in MS-DOS

Other technically respected publications like *Byte* and *Dr. Dobb's* have similar praise for **The Software Toolworks' $49.95** full featured **'C' compiler** for CP/M® and HDOS with:

- I/O redirection
- command line expansion
- execution trace and profile
- initializers
- Macro-80 compatability
- ROMable code
- and much more!

"We bought and evaluated over $1500 worth of 'C' compilers . . . C/80 is the one we use."

— Dr. Bruce E. Wampler
Aspen Software
author of *"Grammatik"*

In reviews published worldwide the amazing **$49.95 C/80** from **The Software Toolworks** has consistently scored at or near the top — even when compared with compilers costing ten times as much!

The optional **C/80 MATHPAK** adds 32-bit floats and longs to the C/80 3.0 compiler. Includes I/O and transcendental function library all for only **$29.95!**

C/80 is only one of 41 great programs each **under sixty bucks.** Includes: LISP, Ratfor, assemblers and over 30 other CP/M® and MSDOS programs.

For your **free** catalog contact:
The Software Toolworks
15233 Ventura Blvd., Suite 1118,
Sherman Oaks, CA 91403 or call 818/986-4885 today!

CP/M is a registered trademark of Digital Research.

Circle **no. 134** on reader service card.

CONTINUING THE TRADITION

DR. DOBB'S JOURNAL ANNOUNCES THE RELEASE OF BOUND VOLUME 8

Every 1983 issue together in one source

Dr. Dobb's Journal
For Users of Small Computer Systems
A Reference Journal for Users of Home Computers
Volume Eight
The People's Computer Company Series

BOUND VOLUME 8
DDJ turns pro. Some of the most powerful, professional programmer's tools ever published in a magazine are in this volume. Jim Hendrix's Small C compiler. Ed Ream's RED screen editor. A microcomputer subset of the Defense Department's official programming language, Ada. C and Forth and 68000 software. Because the magazine increased in size in 1983, this volume is bigger and better that ever.

Vol. 1 1976
The material brought together in this volume chronicles the development in 1976 of Tiny BASIC as an alternative to the "finger blistering," front-panel, machine-language programming which was then the only way to do things. This is always pertinent for the bit crunching and byte saving, language design theory, home-brew computer construction and the technical history of personal computing.
Topics include: Tiny BASIC, the (very) first word on CP/M, Speech Synthesis, Floating Point Routines, Timer Routines, Building an IMSAI, and more.

Vol. 2 1977
1977 found DDJ still on the forefront. These issues offer refinements of Tiny BASIC, plus then state-of-the-art utilities, the advent of PILOT for microcomputers and a great deal of material centering around the Intel 6080, including a complete operating system. Products just becoming available for reviews were the H-8, KIM-1, MITS BASIC, Poly Basic, and NIBL.
Articles are about Lawrence Livermoore Lab's BASIC, Alpha Micro, String Handling, Cyphers, High Speed Interaction, I/O, Tiny Pilot & Turtle Graphics, many utilities, and even more.

Vol. 3 1978
The microcomputer industry entered into its adolescence in 1978. This volume brings together the issues which began dealing with the 6502, with mass-market machines and languages to match. The authors began speaking more in terms of technique, rather than of specific implementations; because of this, they were able to continue laying the groundwork industry would follow. These articles relate very closely to what is generally available today.
Languages covered in depth were SAM76, Pilot, Pascal, and Lisp, in addition to RAM Testers, S-100 Bus Standard Proposal, Disassemblers, Editors, and much, much more.

Vol. 4 1979
This volume heralds a wider interest in telecommunications, in algorithms, and in faster, more powerful utilities and languages, innovation is still present in every page, and more attention is paid to the best ways to use the processors which have proven longevity—primarily the 8080IZ80, 6502, and 6800. The subject matter is invaluable both as a learning tool and as a frequent source of reference.
Main subjects include: Programming Problems/ Solutions, Pascal, Information Network Proposal, Floating Point Arithmetic, 8-bit to 16-bit Conversion, Psuedo-random Sequences, and Interfacing a Micro to a Mainframe—more than ever!

Vol. 5 1980
All the ground-breaking issues from 1980 in one volume! Systems software reached a new level with the advent of CP/M, chronicled herein by Gary Kildall and others (DDJ's all-CP/M issue sold out within weeks of publication). Software portability became a subject of greater import, and DDJ published Ron Cain's immediately famous Small-C compiler—reprinted here in full.
Contents include: The Evolution of CP/M, a CP/M-Flavored C Interpreter, Ron Cain's C Compiler for the 8080. Further with Tiny BASIC, a Syntax-Oriented Compiler Writing Language, CP/M to UCSD Pascal File Conversion, Run-time Library for the Small-C Compiler and, as always, even more!

Vol. 6 1981
1981 saw our first all-FORTH issue (now sold out), along with continuing coverage of CP/M, small-C, telecommunications, and new languages. Dave Cortesi opened "Dr. Dobb's Clinic" in 1981, beginning one of the magazine's most popular features.
Highlights: Information on PCNET, the Conference Tree, and The Electric Phone Book, writing your own compiler, a systems programming language, and Tiny BASIC for the 6809.

Vol. 7 1982
In 1982 we introduced several significant pieces of software, including the RED text editor and the Runic extensible compiler, and we continue to publish utility programs and useful algorithms. Two new columns, The CP/M Exchange and The 16-Bit Software Toolbox, were launched, and we devoted special issues to FORTH and telecommunications. Resident Intern Dave Cortesi supplied a year of "Clinic" columns while delivering his famous review of JRT Pascal and writing the first serious technical comparison of CP/M-86 and MSDOS. This was also the year we began looking forward to today's generation of microprocessors and operating systems, publishing software for the Motorola 68000 and the Zilog Z8000 as well as Unix code. And in December, we looked beyond, in the provocative essay, "Fifth-generation Computers."

Complete your reference library. Buy the entire set of Dr. Dobb's Journals from 1976 through 1983, Bound Volumes 1–8, for $195.00. That's $34.00 off the combined individual prices—a savings of almost 15%!

YES! ☐ Please send me the following Volumes of **Dr. Dobb's Journal.**
Payment must accompany your order.
Please charge my: ☐ Visa ☐ MasterCard ☐ American Express
I enclose ☐ Check/money order
Card # ______________ Expiration Date ______
Signature ______________
Name ______________ Address ______________
(please, no P.O. Boxes)
City ________ State ____ Zip ________
Mail to Dr. Dobb's Journal, 2464 Embarcadero Way, Palo Alto, CA 94303

Allow 6 to 12 weeks for delivery.

Vol. 1	______	x	$27.75 =	______
Vol. 2	______	x	$27.75 =	______
Vol. 3	______	x	$27.75 =	______
Vol. 4	______	x	$27.75 =	______
Vol. 5	______	x	$27.75 =	______
Vol. 6	______	x	$27.75 =	______
Vol. 7	______	x	$30.75 =	______
Vol. 8	______	x	$31.75 =	______
All 8	______	x	$195.00 =	______

Sub-total $ ______
California residents add applicable sales tax____% ______
Postage & Handling Must be Included with order.
Please add $1.75 per book in U.S. ($4.25 each surface mail outside U.S. Foreign Airmail rates available on request.)
______
TOTAL $ ______

3108

DDJ Classifieds

Software

TECMAR GRAPHICS LIBRARY

TEKMAR lets you do high-res graphics on your TECMAR Graphics Master. Features windowing, viewporting, clipping, axis rotation. Similar to Tekronix graphics. Includes screen dump/restore. Epson screen print, support for HP and Western Graphtec plotters. Includes three curve-fitting programs and graphics application SOURCE CODE. Requires MS-FORT 3.20, or Lahey F77L.
Price: $195.
ADVANCED SYSTEMS CONSULTANTS,
18653 Ventura Boulevard, Suite 351,
Tarzana, California 91356
(818) 407-1059

DBase II User—Converting to "C"
Try the dBx translation system— from DBase II to quality "C"; incl. translator, screen handler, & sort (w/source)— uses any file handler. For MSDOS, XENIX & UNIX Sys 5.
Desktop AI
Box 640
Norwalk, CT 06856

Software

DBase II User—Converting to "C"?
Try the dBx translation system—from DBase II to quality "C"; incl. translator, screen handler, & sort (w/source)— uses any file handler. For MS-DOS, XENIX & UNIX sys 5.
Desktop AI
Box 640
Norwalk, CT 06856

SOFTWARE SENTINEL
A hardware key that prohibits unauthorized use of software. Its benefits: Unlimited backup copies; unbreakable; site licensing control; no floppy required with hard disk; transparent; pocketsize. Evaluation Kit available. PC compatible.
Rainbow Technologies, Inc.
17971 Skypark Circle, Suite E
Irvine, CA 92714 (714) 261-0228

Utility

****Pascal's Friend****
PASCAL'S FRIEND v. II contains source code for use with IBM PC Turbo Pascal: 1-2-3 style menu routines, keyboard handling, save and restore screens, read disk directory or any track and sector, system clock and calendar routines, write strings in any attribute, DOS funcion calls and MORE!
J. S. Computing
815 N. 12th St.
Suite 5
Allentown, PA 18102
(215) 821-9020

Recruiting

Personal Computer Owners
CAN EARN $1000 TO $5000
monthly selling simple services performed by their computer. Work at home in spare time. Get free list of 100 best services to offer.

Write:
C.I.L.D.G.
P. O. Box 60369
San Diego, CA 92106-8369

Hardware

DIAL & ORDER
24 HR. MODEM LINE
(408)425-1603
Specializing in technical/reference books and computer supplies.
Wise Dog Computerbooks
1310 Fair Ave, Santa Cruz, CA 95060

Dr. Dobb's Journal Subscription Problems?

No Problem!
Give us a call and we'll straighten it out. Today.

Outside CA: 800-321-3333
Inside CA: 619-485-9623 or 6947

Dr. Dobb's Journal
is pleased to announce the
DDJ Classifieds

RATES: DISPLAY ADVERTISERS: Price per column inch $100. Ad must run in 3 consecutive issues.
LINE ADVERTISERS: Price per line $12 (35 characters per line including spaces). Minimum of 5 lines, 3 consecutive issues. Add $3 per line for boldface type. Add $25 if a background screen is desired.
DISCOUNTS: Frequency discounts for 6 consecutive ads (less 7%) and for 12 consecutive ads (less 15%).
MECHANICAL REQUIREMENTS: Camera ready art or typewritten copy only (phone orders excepted).
Display: Specify desired size, include $20 if reduction is required. Line: Specify characters in boldface, caps. Allow 6 weeks for publication.
PAYMENTS: Full payment in advance. Check, money order, Visa, M/C, AmEx.

Run this ad in ________ issues

Size ________ col x ________ inches or 1 col x ________ lines

Payments by: ________ check ________ m/o ________ Visa ________ M/C ________ AmEx

Card # ____________________ Exp Date ________

Signature ____________________

Print Name ____________________

Company ____________________

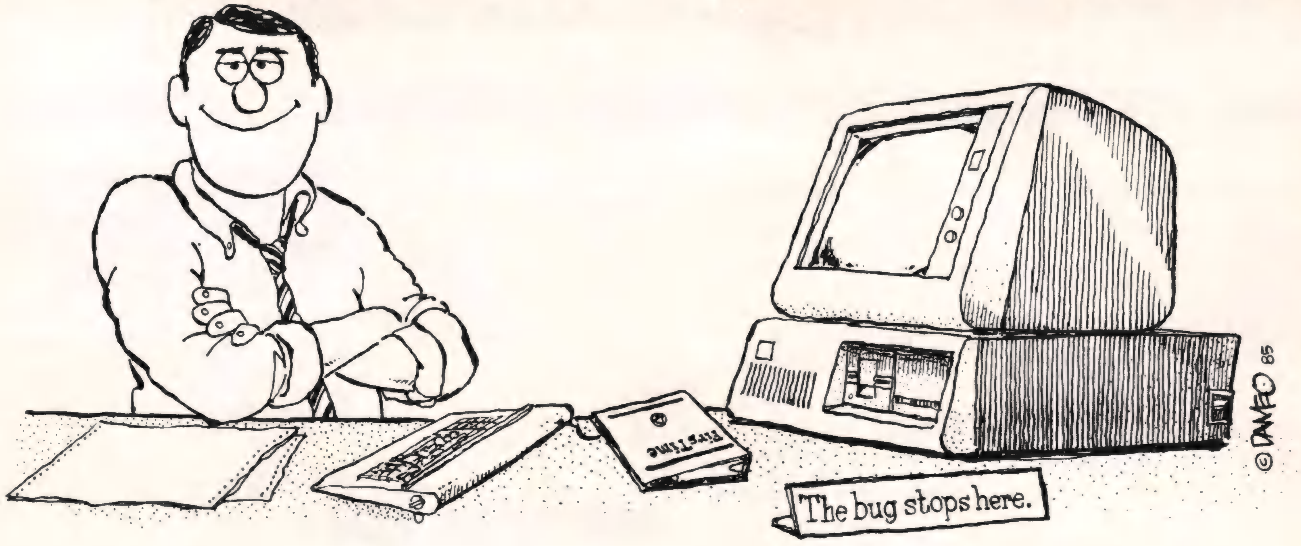
Address ____________________

City ____________________ State ________ Zip ________

Phone ____________________ Current Subscriber ________

Mail to or phone Shawn Horst, DDJ Classifieds, 2464 Embarcadero Way, Palo Alto, CA 94303 (415) 424-0600

The First Idea-Processor For Programmers.



FirsTime™

Has features no other editor has.

- Fast program entry through single keystroke statement generators.
- Fast editing through syntax oriented cursor movements.
- Dramatically reduced debugging time through immediate syntax checking.
- Fast development through unique programmer oriented features.
- Automatic program formatter.

FirsTime is a True Syntax Directed Editor.

As the world's most advanced syntax-directed editor, FirsTime lets you work with ideas by taking care of the low-level syntax details of your program. For example, you can generate complete statement skeletons with one keystroke. Move the cursor from one procedure to the next with one keystroke. Type in code, and it's instantly formatted (you specify the rules). Type an error, and FirsTime warns you immediately. You can continue working if you wish. Later, you can use the search-for-error command to find the error and fix it.

FirsTime Has Thorough Error Checking.

FirsTime not only checks your syntax, but also semantics. FirsTime identifies:
- Undefined variables, types and constants.
- Assignment statements with type mismatches.
- Errors in include files and macro expansions.

FirsTime Lets You Work With Ideas.

The *Zoom command* gives you a top down view of your program logic.

The *View macro command* shows the expansions of a C macro while in the editor.

The *View include file command* instantly shows you the contents of an include file.

The *Transform command* allows you to change a statement to another similar statement, for example, a *FOR* to an equivalent *WHILE*.

The *Search for next error* command allows you to find errors throughout your program.

The *Cursor movement commands* let you traverse your program by logical elements, not just characters.

FirsTime Works With Existing Files.

FirsTime works with standard ASCII files, so you can edit any existing source files.

To Order Call: (201) 741-8188 **or write:**

SPRUCE TECHNOLOGY CORPORATION

P.O. Box 7948
Shrewsbury, NJ 07701

FirsTime for Turbo Pascal	**$ 74.95**
FirsTime for dBase III	**$125.00**
FirsTime for MS–Pascal	**$245.00**
FirsTime for C	**$295.00**

In Germany, Austria and Switzerland contact:
Markt & Technik Software Verlag
Munchen, W. Germany
(089) 4613-0

FirsTime is a trademark of Spruce Technology Corporation • MS–DOS is a trademark of Microsoft Corporation • IBM is a trademark of International Business Machines Inc. • Turbo Pascal is a trademark of Borland International • dBase III is a trademark of Ashton-Tate.

Circle no. **164** on reader service card.

Tools That Make Your Job Easier

For PCDOS/MSDOS (2.0 and above/128K) • IBM PC/Compatibles, PC Jr., Tandy 1000/1200/2000, & others
For CPM80 2.2/3.0 (Z80 required/64K) • 8" SSSD, Kaypro 2/4, Osborne I SD/DD, Apple II, & others

MIX EDITOR

Programmable, Full/Split Screen Text Processor

Introductory Offer **29.95**

Great For All Languages

A general purpose text processor, the MIX Editor is packed with features that make it useful with any language. It has auto indent for structured languages like Pascal or C. It has automatic line numbering for BASIC (255 character lines). It even has fill and justify for English.

Split Screen

You can split the screen horizontally or vertically and edit two files simultaneously.

Custom Key Layouts

Commands are mapped to keys just like WordStar. If you don't like the WordStar layout, it's easy to change it. Any key can be mapped to any command. You can also define a key to generate a string of characters, great for entering keywords.

Macro Commands

The MIX Editor allows a sequence of commands to be executed with a single keystroke. You can define a complete editing operation and perform it at the touch of a key.

Custom Setup Files

Custom keyboard layouts and macro commands can be saved in setup files. You can create a different setup file for each language you use.

MSDOS Features

Execute any DOS command or run another program from inside the editor. You can even enter DOS and then return to the editor by typing exit.

MIX C COMPILER

Full K&R Standard C Language
Unix Compatible Function Library

Introductory Offer **39.95**

Complete & Standard

MIX C is a complete and standard implementation of C as defined by Kernighan and Ritchie. Coupled with a Unix compatible function library, it greatly enhances your ability to write portable programs.

The Best C Manual

MIX C is complemented by a 400 page manual that includes a tutorial. It explains all the various features of the C language. You may find it more helpful than many of the books written about C.

Fast Development

MIX C includes a fast single pass compiler and an equally fast linker. Both are executed with a simple one line command. Together they make program development a quick and easy process.

Fast Execution

The programs developed with MIX C are fast. For example, the often quoted prime number benchmark executes in a very respectable 17 seconds on a standard IBM PC.

Standard Functions

In addition to the functions described by K&R, MIX C includes the more exotic functions like *setjmp* and *longjmp*. Source code is also included.

Special Functions

MIX C provides access to your machine's specific features through BDOS and BIOS functions. The CHAIN function lets you chain from one program to another. The MSDOS version even has one function that executes any DOS command string while another executes programs and returns.

Language Features

- Data Types: char, short, int, unsigned, long, float, double (MSDOS version performs BCD arithmetic on float and double-no roundoff errors)
- Data Classes: auto, static, extern, register
- Struct, Union, Bit Fields (struct assignment supported)
- Typedef, Initialization
- All operators and macro commands are supported

30 DAY MONEY BACK GUARANTEE

Orders Only: **Call Toll Free 1-800-523-9520**, (Texas only **1-800-622-4070**)

MIX Editor ____ $29.95 + shipping ($5 USA/$10 Foreign) Texas residents add 6% sales tax
MIX C ____ $39.95 + shipping ($5 USA/$25 Foreign) Texas residents add 6% sales tax

Visa ____ MasterCard ____ Card # ______________________ Exp. Date ________

COD ____ Check ____ Money Order ____ Disk Format ______________________

Computer ______________ Operating System: MSDOS ____ PCDOS ____ CPM80 ____

Name ______________________

Street ______________________

City/State/Zip ______________________

Country ______________________

Phone ______________________

MIX software
2116 E. Arapaho
Suite 363
Richardson, Tx 75081

Dealer Inquiries Welcome
Call (214) 783-6001

MSDOS is a trademark of Microsoft PCDOS is a trademark of IBM CPM80 is a trademark of Digital Research WordStar is a trademark of MicroPro

D

The Best Source Debugger for C

Interactive testing: enter any C expression, statement, or function call, and it is immediately executed and the result displayed. **Direct execution** allows fast and thorough testing, makes learning C a snap.

Run-time checking: execution stops upon exception, and source code displayed. Exceptions include array reference bounds, stack overflow arithmetic or floating point error, etc. Pointers are checked for null or out of range values.

Breakpoints: *any* number of breakpoints can be set *anywhere* in your program; breakpoints are set with screen editor, not by line numbers. Breakpoints may be conditional. Single-step by statement. Interrupt execution from keyboard. Breakpoint, exception, or interruption is always shown with source code. Examine and modify data, look at stack history. Even change your program and then resume execution!

Lint-like **Compile-time checking:** argument number and sizes are checked for consistency. Never mismatch source and object code.

The best feature of all: the *fastest* C interpreter is right there when you're debugging. Make changes in seconds with the integrated screen editor. Test the changes immediately, running your program at compiled speed. Save source code for your favorite compiler, or make stand-alone executable programs. Nothing else is needed. ***Instant-C* is the fastest way to get working, fully debugged C programs available today.**

"We sincerely feel that ***Instant-C*** can have a major positive impact on programmer productivity." *Computer Language*, Feb. 85, pp. 82-83.

Instant-C is only $495. Money back for any reason in first 31 days.

Rational
Systems, Inc.
(617) 653-6194
P.O. Box 480
Natick, MA 07160

Circle **no. 7** on reader service card.

ADVERTISER INDEX

Reader Service No.	Advertiser	Page No.
4	Advent Products, Inc.	64-65
*	Alcor/Mix Systems	127
5	Amanuensis	48
16	Arity Corporation	37
12	Automata Design Systems	84
29	B.C.C., Inc.	63
8	Blaise Computing	27
14	Borland International	C-4
126	Boston Software Works	119
17	C User's Group	119
57	C Ware	61
11	Cardinal Pointe, Inc.	119
19	Chalcedony	13
24	Cogitate, Inc.	119
18	Computer Control Systems	107
22	Computer Helper Industries, Inc.	92
96	Computer Innovations	117
32	Creative Programming	12
28	D&W Digital	28
122	DDJ Classified Advertising	125
21	DMCQ	85
51	DPMA	77
31	Data Base Decisions	97
33	Digital Research Computers	121
53	Earth Computers	50
168	Eclipse Systems	18
35	Ecosoft, Inc.	58
13	Edward K. Ream	48
138	Essential Software	55
30	Everest Solutions	C-2
37	Faircom	99
40	Fox Software, Inc.	56
*	Gimpel Software	21
*	Gimpel Software	40
45	Golemics	18
43	Greenleaf Software, Inc.	66
6	Guidance Software	83
26	Hallock Systems Consultants	32
44	Harvard Softworks	100
48	HiSoft	98
23	HiTech Equipment Corp.	85
46	Illyes Systems	36
2	Inetco Systems	61
61	Institute for Applied Forth Tech.	87
56	Integral Quality	31
15	Integrand Research Corp.	47
73	Intelliware, Inc.	109
55	Laboratory Microsystems Inc.	43
36	Lattice, Inc.	68
60	Liberty Bell Publishing	62
125	Lifeboat Associates	60
9	Logique	91
83	Manx Software	59
74	Martian Technologies	112
67	Media Cybernetics	29
84	Megamax, Inc.	81
70	Micro Motion	46
64	Microprocessors Unlimited	119
41	MicroSmith	119
*	Microsoft	1
72	Microsoft Press	67
50	Microtec Research	7

Reader Service No.	Advertiser	Page No.
65	Miller Microcomputer Services	16
66	Mitek	23
128	Morgan Computing Company	115
104	Mullen Computer Products	93
79	Mystic Canyon Software	17
106	Next Generation Systems	26
39	Overland Data, Inc.	73
108	P.C. Brand	94-95
25	Pascom Computing	C-3
110	Performance Microproducts	112
76	Personal Tex Inc.	72
47	Phoenix Computer Products	2-3
91	Phoenix Computer Products	15
69	Plu Perfect Systems	86
71	Poor Person Software	81
99	Programmer's Shop	51
7	Rational Systems, Inc.	128
49	Relational Database Systems	53
80	Revasco	109
78	SLR Systems	69
20	Samkhya	112
114	Seidl Computer Engineering	110
85	SemiDisk Systems	108
86	Shaw American Technologies	49
124	Shaw Laboratories	35
63	Soft Advances	54
118	SoftCraft, Inc.	19
88	Softaid, Inc.	49
120	Softfocus	93
140	Software Horizons, Inc.	30
134	Software Toolworks	123
92	Softway, Inc.	103
75	Programmer's Shop	79
102	Solution Systems	113
100	Sota Computing Systems	34
136	Speech Technology	98
59	Speedware	110
164	Spruce	126
142	Starlight Forth Systems	111
10	Sunny Hill Software	119
144	Sunset Technology	111
162	Systems Guild	122
104	Systems Management Assoc.	33
146	The Dragon Group	98
148	The Librarian	92
81	Turbo Power Software	38
150	US Software	71
152	Ubiquitous Systems	119
77	UniPress Software	41
154	Vance Info Systems	87
27	Vermont Creative Software	88-89
156	Vesta Technologies	119
112	Wendin, Inc.	9
116	Wizard Systems	90
160	Zedcor	11
*	*DDJ* Back Issues	116
*	*DDJ* Bound Volume	124
*	*DDJ* Source Book	75
*	*DDJ* Subscription	80
*	*DDJ* Toolbook/Disks	104-105
*	*DDJ* Fatten Your Mac reprints	93
*	*DDJ* C Toolbook/Bound Volume	32a

* This advertiser prefers to be contacted directly: see ad for phone number.

Advertising Sales Offices

East Coast
Walter Andrzejewski (617) 567-8361

Northern California/Northwest
Lisa Boudreau (415) 424-0600

Midwest/West Central
Michele Beaty (317) 875-0557

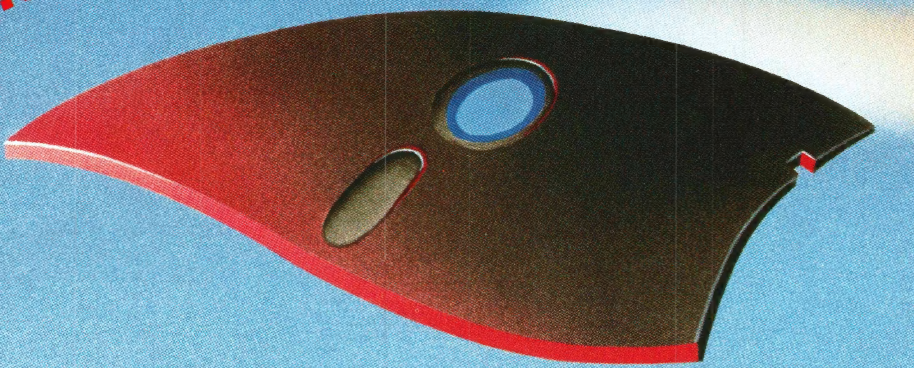
Southern California/AZ, NM
Beth Dudas (714) 643-9439

Advertising Director
Shawn Horst (415) 424-0600

YOU'VE GOT THE BEST PASCAL COMPILER!
NOW — GET THE BEST UTILITY!

TURBO SCREEN™

$49.95



NEVER AGAIN WRITE SOURCE CODE FOR SCREEN DISPLAYS!

*If you **LIKE** Turbo Pascal*, you'll **LOVE** TURBO SCREEN™!*

Tired of writing line after line of source code just to create effective screen displays and error-proof data handling? Then use TURBO SCREEN's Editor to create the screens, the Collator to define a list of screens . . . and then **relax** for a few seconds **while the Generator writes the code!**

TURBO SCREEN™

- 100 Fields per screen, and up to 80 screens in your application.
- One screen or eighty, the size of your program doesn't change.
- I/O field types of:
  Real, Integer, String, Character, Boolean.
- "Bullet-proof" data entry.
- Create Window-Style overlays or Full-screen pictures in CP/M*, MS-DOS*, or PC-DOS.
- Supports video attributes for your terminal. And YES, if you have an RGB monitor, you can create screens in COLOR on your IBM PC or true compatible.
- A SINGLE LINE of source code invoking TURBO SCREEN'S "display" procedure controls:
  —picture selection
  —output to screen, printer, or disk
  —I/O field update
- TURBO SCREEN™ is completely menu-driven and includes a built-in Screen Editor, Collator, and Generator, each called up with a single keystroke!
- ADVANCED EDITOR:
  —Turbo Pascal*—like commands include:
    —Block commands for copy, fill, exchange, erase.
    —Draw lines in any direction with any character.
    —Supports IBM color monitor and graphics characters.
- FAST—Generates code for 20 screens in about 60 seconds!
- DISK UTILITIES built-in:
  —directories
  —erase files
- REQUIRES:
  —Turbo Pascal any version
  —80x24 or larger video screen
- AVOID software "bottlenecks!"

PASCOM COMPUTING
23611 Chagrin Blvd., Suite 101
Cleveland, Ohio 44122

Start letting TURBO SCREEN™ write your I/O source code today!
VISA MasterCard ONLY — Call TOLL-FREE: 1-800-243-1849
Inside Ohio call 1-216-292-8745 (Lines Open 24 hours, 7 days)

Check ________	TURBO SCREEN™ package	$49.95
Money Order ________	Plus Ship. (UPS)	5.00
Visa ________	Total	$54.95
Master Card ________		

Card # ____________________

Exp. Date: ____________________

Computer System: ________ 8-bit ________ 16-bit
Operating System: ________ CP/M80 ________ PC-DOS
________ CP/M86 ________ MS-DOS
Computer Model: ____________________ Disk Format: ____________________
Name: ____________________
Address: ____________________
City: ____________________ State ________ Zip ________
Telephone: ____________________

Ohio residents add 6½% sales tax. Outside U.S.A. add $20.00

U.S. Dealer Inquiries Welcome.

Circle **no. 25** on reader service card.

*Turbo Pascal is a trademark of Borland International. IBM is a trademark of International Business Machines. MS-DOS is a trademark of Microsoft. CP/M is a trademark of Digital Research.

60-DAY MONEY BACK GUARANTEE

Borland's SuperKey And SideKick Work So Well Together, You'll Hardly Work At All.

AN UNBEATABLE TEAM AT AN UNBEATABLE PRICE! We've teamed the best with the best to make the greatest. The best keyboard enhancer, SuperKey®. The best desktop organizer, SideKick®. The dynamic duo working hand in hand to let you do many different things at once. A way that cuts down the keystrokes, so you're working instead of just typing. A way that wasn't possible until we paired the electronic wizardry of SuperKey with the practical efficiency of SideKick. SuperKey brings the magic. SideKick does the details. The "S-Team" works beautifully together because we designed them that way.

GET SUPERKEY AND SIDEKICK TODAY and you'll have an unbeatable team at an unbeatable price — and a $15 rebate back in the mail.

SIDEKICK INCLUDES: * Calculator * Notepad * Auto dialer & phone directory * ASCII table * Perpetual calendar & datebook * Help window * Full-screen editor with word-wrap, paragraph editing and much much more. (Chances are that once you have SuperKey and SideKick working together for you, you'll never need to use a word-processing program again).

SUPERKEY INCLUDES: *Encryption to keep confidential files confidential * Programmable keys that let you turn a thousand keystrokes into one * Keyboard lock * Automatic turn-off of your screen after a pre-set time so the expensive phosphorus in your monitor's screen isn't etched or ruined * Secret Password protection and more.

IF YOU USE SIDEKICK, YOU NEED SUPERKEY. BECAUSE SUPERKEY AND SIDEKICK CAN MAKE YOUR DAY GO SOMETHING LIKE THIS:

8:00 am. You got to work on time, despite the 44-mph turkey ahead of you in the fast lane. It's spreadsheet time. You hit one key. Lotus 1-2-3 (or whatever) is up and running. (One key, because SuperKey has recorded all the CD\123 <ENTER >123< ENTER> <ENTER> / F <ENTER> R <ENTER> SALES <ENTER> <PgDn> foolishness and your one keystroke played all that back instantly. One keystroke instead of a minuet).

8:03 am. You're into the spreadsheet. Phone rings. You kick in SideKick's Notepad—without leaving your spreadsheet. You talk. You listen to Frank. You make notes that tell you that Frank is upping the numbers from yesterday's order and he needs a new price and delivery date. He wants a meeting. Fast, but when? You have SideKick fire up your Calendar. Time agreed and noted—in SideKick's NotePad. Conversation ends. Your spreadsheet is still there.

8:07 am. You're watching the spreadsheet but you're thinking about the new bid you have to figure out. So you have SideKick's Calculator pulled up on the screen—over a small piece of the spreadsheet—which doesn't go away.

8:08 am. SideKick is coming up with new numbers. SuperKey keeps the spreadsheet on a roll. Satisfied with the numbers, you have SideKick auto-dial Frank's number. Talk. Talk. Hang up.

8:09 am. Spreadsheet about done. You're watching it, but thinking about what Frank just said on the phone. He liked your numbers. He ordered. He said, "That was fast. We won't need that meeting. (SideKick cancels it from your Calendar). And he also said, "How did you get all that done so quickly?" And you said, "I've got a couple of new guys working for me."

IF YOU DON'T USE SIDEKICK YET, YOU GET THEM BOTH AND FOR A LIMITED TIME, A $15.00 CASH-BACK! Because SuperKey and SideKick are so compatible, we let them move in together. Into their own blister-pack. With the $15.00 cash-back coupon and the manuals. Which is what you get for $139.90 instead of the usual $154.90. You need to fill-in the cash-back coupon, along with your registration cards and proof of purchase, and mail it back. We'll rush you your $15.00 rebate right away. Rebate offer ends March 31, 1986. (PS: You can still buy SuperKey and SideKick separately. SuperKey ***$69.95***, SideKick ***$84.95***. Not copy-protected.)

THE CRITIC'S CHOICE

"If you want the ultimate in sophistication, you won't find anything finer on the battlefield right now than Borland's SuperKey and SideKick combo." **LEON A. WORTMAN, InfoWorld**

BORLAND

4585 Scotts Valley Drive, Scotts Valley CA 95066
Phone (408) 438-8400 Telex 172373

NOT COPY-PROTECTED

SuperKey SIDEKICK

SuperKey and SideKick require DOS version 2.0 or later.
Software for your IBM®PC, XT, AT, 3270, PCjr and true compatibles.

Available at better dealers nationwide. Call (800) 556-2283 for the dealer nearest you. To order by Credit Card call (800) 255-8008, CA (800) 742-1133.

Yes, I want the Best!

	Quantity
The SideKick-SuperKey Team **$154.90** $15 cash-back ONLY when purchased together. Both come not copy protected.	______
SuperKey **$69.95**	______
SideKick **$54.95** (copy protected)	______
SideKick **$84.95** (not copy-protected)	______

Subtotal ______
(CA res. add 6% tax) ______
Amount Enclosed: ______
Payment VISA MC BankDraft Check
Credit Card Exp. Date ____/____
Card # ______

$15⁰⁰ Cash Back

SuperKey only—Outside USA and Canada: Due to US Government restrictions of export or technology, purchase of SuperKey (without DES) can only be made through authorized distributors.

SideKick only: PCjr requires the not copy-protected version.

$15 cash-back rebate upon receipt of signed license agreement and cash-back coupon. Your $15 cash-back rebate includes tax when applicable. These prices include shipping to all U.S. cities. All foreign orders add $10 per product ordered.

Name: ______
Shipping Address: ______
______
City: ______
State: ______ Zip: ______
Telephone: ______

F11

COD's and Purchase Orders WILL NOT be accepted by Borland. California residents: add 6% sales tax. Outside USA: add $10 and make payment by bank draft, payable in US dollars drawn on a US bank.

Circle **no. 14** on reader service card.